4-2005

Structure Preserving Algorithms for Computing the Symplectic Singular Value Decom Position

Archara Chaiyakarn
Western Michigan University

Follow this and additional works at: http://scholarworks.wmich.edu/dissertations

Part of the Applied Mathematics Commons, and the Mathematics Commons

Recommended Citation
Dissertations, 1022.
http://scholarworks.wmich.edu/dissertations/1022

This Dissertation-Open Access is brought to you for free and open access by the Graduate College at ScholarWorks at WMU. It has been accepted for inclusion in Dissertations by an authorized administrator of ScholarWorks at WMU. For more information, please contact maira.bundza@wmich.edu.
STRUCTURE PRESERVING ALGORITHMS
FOR COMPUTING THE SYMPLECTIC
SINGULAR VALUE DECOMPOSITION

by

Archara Chaiyakarn

A Dissertation
Submitted to the
Faculty of The Graduate College
in partial fulfillment of the
requirements for the
Degree of Doctor of Philosophy
Department of Mathematics

Western Michigan University
Kalamazoo, Michigan
April 2005
STRUCTURE PRESERVING ALGORITHMS
FOR COMPUTING THE SYMPLECTIC
SINGULAR VALUE DECOMPOSITION

Archara Chaiyakarn, Ph.D.

Western Michigan University, 2005

In this thesis we develop two types of structure preserving Jacobi algorithms for computing the symplectic singular value decomposition of real symplectic matrices and complex symplectic matrices. Unlike general purpose algorithms, these algorithms produce symplectic structure in all factors of the singular value decomposition.

Our first algorithm uses the relation between the singular value decomposition and the polar decomposition to reduce the problem of finding the symplectic singular value decomposition to that of calculating the structured spectral decomposition of a doubly structured matrix. A Jacobi-like method is developed to compute this doubly structured spectral decomposition.

The second algorithm is a one-sided Jacobi method that directly computes the structured singular value decomposition of real or complex symplectic matrices.

Numerical experiments show that our algorithms converge quadratically. Furthermore, the number of sweeps needed for convergence is favorable when compared to Jacobi-like algorithms for other structured matrices.
INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.
I am greatly indebted to my advisor, Dr. Niloufer Mackey for her valuable time, for her continuous support, for her guidance throughout the process of the completion of this thesis. Her wide knowledge and her logical way of thinking have been a great value for me. Besides being an excellent advisor, she is as close as a relative and a good friend to me.

I would like to thank all the members of my committee — Dr. Niloufer Mackey, Dr. Dennis D. Pence, Dr. Clifton E. Ealy Jr., and Dr. Hongguo Xu for providing me with valuable comments and suggestions on early versions of this thesis.

I owe thanks to Dr. Niloufer Mackey, Dr. Annegret Paul, and Dr. Clement Burns for letting me use their computers to run numerical experiments. I also want to thank systems analyst Mr. John D. Tucker for his valuable assistance in running unattended jobs on the Sun network.

Thanks also go to all my friends in the Department of Mathematics, particularly Dr. Kirsty Eisenhart, Dr. Theresa Grant, and Daniela Hernandez whose boundless support and humor have helped me through many of the most difficult days.

Thanks to the faculty and staff at the Department of Mathematics for their devotion to their students.

I also want to thank my parents, who taught me the value of hard work by their own example. I would like to share this moment of happiness with my
father, mother, sisters and brother. They gave me enormous support during my education.

Finally, I would like to give my very special thanks to Pariwat Pacheenburawana, for his encouragement and understanding.

Archara Chaiyakarn
Table of Contents

Acknowledgments ii
List of Tables vi
List of Figures viii
Notation x

1 Introduction 1
1.1 Symplectic Groups ................................................................. 3
1.1.1 Symplectic Unitary Groups .............................................. 5
1.2 Symplectic Singular Value Decomposition ............................... 6

2 Structured Tools 9
2.1 The Classical Jacobi Method .................................................... 9
2.2 Real Symplectic Plane Rotations .............................................. 14
2.2.1 Givens-like Action ............................................................. 16
2.2.2 Jacobi-like Action ............................................................. 19
2.3 Real Symplectic Double Rotations .......................................... 21
2.3.1 Direct Sum Embedding ....................................................... 21
2.3.2 Concentric Embedding ....................................................... 23
2.4 Real 4 × 4 Symplectic Orthogonals ......................................... 27
2.5 Real Symplectic Householders ................................................ 29

3 Real Symplectic Bird Form 32
3.1 Via Symplectic Householders .................................................. 33
3.2 Via 4 × 4 Symplectic Givens ................................................... 37

4 Computing the Real Symplectic SVD: Using the Polar Decomposition 41
4.1 4 × 4 Case ................................................................................. 42
4.2 Sweep Designs for 2n × 2n Case .............................................. 50
4.2.1 Strategy S1 ........................................................................... 51
4.2.2 Strategy \( S_2 \) ......................................................... 55
4.2.3 Strategy \( S_3 \) ......................................................... 57
4.2.4 Strategy \( S_4 \) ......................................................... 59
4.2.5 Strategy \( S_5 \) ......................................................... 61
4.2.6 Strategy \( S_6 \) ......................................................... 63
4.3 Description of the Algorithms ........................................ 66
4.4 Numerical Experiments .............................................. 71
  4.4.1 Structured Spectral Decomposition ....................... 72
  4.4.2 Structured SVD ................................................. 85

5 Computing the Real Symplectic SVD: One-sided Jacobi 94
  5.1 General One-sided Jacobi Method ........................... 94
  5.2 Adaptation to Real Symplectic Matrices ................. 96
  5.3 Numerical Experiments ......................................... 102

6 Computing the Complex Symplectic SVD 110
  6.1 Using the Polar Decomposition .............................. 110
    6.1.1 Structured Spectral Decomposition of Complex Symplectic
           Hermitian Matrices ........................................ 111
    6.1.2 Numerical Experiments .................................. 124
  6.2 One-sided Jacobi Method ..................................... 132
    6.2.1 Numerical Experiments .................................. 140

7 Concluding Remarks ................................................. 146

Bibliography ............................................................. 149
List of Tables

4.1 Average number of sweeps needed for 30 × 30 matrices ............ 72
4.2 \textit{reloff.tol} = 10^{-14} \times 10^{m} \times \|A\|_F for the specified condition numbers 73
4.3 Average number of sweeps vs. condition numbers: \mathcal{S}_4 ........ 74
4.4 Average number of sweeps vs. condition numbers: \mathcal{S}_5, \mathcal{S}_6 .... 74
4.5 Statistical data for the number of sweeps ............................. 75
4.6 Statistical data for the total number of iterations ................... 77
4.7 The maximum cost per sweep for \mathcal{S}_4, \mathcal{S}_5 and \mathcal{S}_6 ............ 79
4.8 Strategy \mathcal{S}_{4s} ......................................................... 80
4.9 Strategy \mathcal{S}_{4rs} ....................................................... 80
4.10 Strategy \mathcal{S}_{5s} ......................................................... 80
4.11 Strategy \mathcal{S}_{5rs} ....................................................... 81
4.12 Strategy \mathcal{S}_{6s} ......................................................... 81
4.13 Strategy \mathcal{S}_{6rs} ....................................................... 81
4.14 Accuracy of the eigenvalues 10, 5, 2, \frac{1}{2}, \frac{1}{5}, \frac{1}{10} .......... 82
4.15 Accuracy of the eigenvalues 10, 5, 5, \frac{1}{5}, \frac{1}{10} .................... 82
4.16 Accuracy of the eigenvalues near 3 and \frac{1}{3} ......................... 83
4.17 Accuracy of the eigenvalues 10^4, 10^2, 10, \frac{1}{10}, \frac{1}{10^2}, \frac{1}{10^4} ... 83
4.18 Measuring the non-monotonicity of off(A) when A is a typical real

50 × 50 symplectic symmetric matrix .......................... 84
4.19 Statistical data for real symplectic matrices and their polar factor 87
4.20 Statistical data for the real symplectic SVD: \mathcal{S}_{4s} .............. 88
4.21 Statistical data for the real symplectic SVD: \mathcal{S}_{4rs} .............. 88
4.22 Statistical data for the real symplectic SVD: \mathcal{S}_{5s} .............. 89
4.23 Statistical data for the real symplectic SVD: \mathcal{S}_{5rs} .............. 89
4.24 Statistical data for the real symplectic SVD: \mathcal{S}_{6s} .............. 90
4.25 Statistical data for the real symplectic SVD: \mathcal{S}_{6rs} .............. 90
4.26 Accuracy of the singular values 10, 5, 2, \frac{1}{2}, \frac{1}{5}, \frac{1}{10} .......... 92
4.27 Accuracy of the singular values near 3 and \frac{1}{3} .................... 92
4.28 Accuracy of the singular values 10^4, 10^2, 10, \frac{1}{10}, \frac{1}{10^2}, \frac{1}{10^4} ... 93
4.29 Accuracy of the singular values near 3, \frac{1}{3}, 5 and \frac{1}{5} ............ 93
5.1 Statistical data for the total number of iterations .................. 104
5.2 One-sided Jacobi algorithm: \mathcal{O}S_4 ................................. 105
5.3 One-sided Jacobi algorithm: \( OS_{4s} \) .................................................. 105
5.4 One-sided Jacobi algorithm: \( OS_5 \) .................................................. 105
5.5 One-sided Jacobi algorithm: \( OS_{5s} \) .................................................. 106
5.6 One-sided Jacobi algorithm: \( OS_6 \) .................................................. 106
5.7 One-sided Jacobi algorithm: \( OS_{6s} \) .................................................. 106
5.8 Accuracy of the singular values 10, 5, 2, \( \frac{1}{2} \), \( \frac{1}{5} \), \( \frac{1}{10} \) .................. 107
5.9 Accuracy of the singular values near 3 and \( \frac{1}{3} \) .................................. 107
5.10 Accuracy of the singular values near 100 and \( \frac{1}{100} \) ....................... 108
5.11 Accuracy of the singular values 10, 10, 2, \( \frac{1}{10} \), \( \frac{1}{100} \), \( \frac{1}{10^2} \) .... 108
5.12 Accuracy of the singular values near 3, \( \frac{1}{3} \), 5 and \( \frac{1}{5} \) ................. 109

6.1 Symplecticity of Hermitian matrices .................................................. 116
6.2 Statistical data for the complex symplectic SVD: \( CS_4 \) ....................... 125
6.3 Statistical data for the complex symplectic SVD: \( CS_{4s} \) ..................... 125
6.4 Statistical data for the complex symplectic SVD: \( CS_5 \) ....................... 126
6.5 Statistical data for the complex symplectic SVD: \( CS_{5s} \) ..................... 126
6.6 Statistical data for the complex symplectic SVD: \( CS_6 \) ....................... 127
6.7 Statistical data for the complex symplectic SVD: \( CS_{6s} \) ..................... 127
6.8 Accuracy of the singular values 5, 5, 3, \( \frac{1}{5} \), \( \frac{1}{3} \) .......................... 131
6.9 Accuracy of the singular values near 3 and \( \frac{1}{3} \) ................................ 131
6.10 Accuracy of the singular values near 3 and 5, 5, \( \frac{1}{5} \), \( \frac{1}{5} \) .................. 132
6.11 Accuracy of the singular values near 3, \( \frac{1}{3} \), 5 and \( \frac{1}{5} \) .................. 132
6.12 Complex One-sided Jacobi algorithm: \( OC_{S_4} \) .............................. 141
6.13 Complex One-sided Jacobi algorithm: \( OC_{S_{4s}} \) ............................ 141
6.14 Complex One-sided Jacobi algorithm: \( OC_{S_5} \) .............................. 141
6.15 Complex One-sided Jacobi algorithm: \( OC_{S_{5s}} \) ............................ 142
6.16 Complex One-sided Jacobi algorithm: \( OC_{S_6} \) .............................. 142
6.17 Complex One-sided Jacobi algorithm: \( OC_{S_{6s}} \) ............................ 142
6.18 Accuracy of the singular values 5, 5, 3, \( \frac{1}{5} \), \( \frac{1}{3} \) .......................... 144
6.19 Accuracy of the singular values near 3 and \( \frac{1}{3} \) ................................ 144
6.20 Accuracy of the singular values 100, 10, 2, \( \frac{1}{100} \), \( \frac{1}{10} \), \( \frac{1}{2} \) ........ 144
6.21 Accuracy of the singular values near 3, \( \frac{1}{3} \), 5 and \( \frac{1}{5} \) .................. 145

vii

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
List of Figures

2.1 Left action: rows $p, n + p$ changed ............................................ 17
2.2 Right action: columns $p, n + p$ changed .................................... 18
2.3 Action by outer rotation $R_1$ on $A$ ............................................. 25
2.4 Action by inner rotation $R_2$ on $A^{(1)}$ ......................................... 26
2.5 Left action by $4 \times 4$ Givens rotation ......................................... 29
2.6 Left action by real symplectic Householder matrix ....................... 30

3.1 Reduction of a real $8 \times 8$ symplectic symmetric matrix to the sym­
plectic bird form using real symplectic Householders and real sym­
plectic plane rotations ............................................................................. 34
3.2 Reduction of a real $8 \times 8$ symplectic symmetric matrix to the sym­
plectic bird form using $4 \times 4$ Givens and $2 \times 2$ Givens .................... 38

4.1 Sweep design for Strategy $S_1$ to reduce a real symplectic symmetric
matrix to matrix with diagonal blocks ................................................. 52
4.2 Reduction of a diagonal block matrix to diagonal form ................... 55
4.3 Sweep design for strategy $S_2$ ...................................................... 56
4.4 Sweep design for strategy $S_3$ ...................................................... 58
4.5 Structured sweep for Strategy $S_4$ ................................................ 59
4.6 Structured sweep for Strategy $S_5$ ................................................ 62
4.7 Structured sweep for Strategy $S_6$ ................................................ 65
4.8 Average number of sweeps vs. condition numbers: $S_4$.
Note that the number of iterations in a full sweep of $S_4$ is $n^2$ .......... 76
4.9 Average number of sweeps vs. condition numbers: $S_5, S_6$.
Note that the number of iterations in a full sweep of $S_5$ or $S_6$ is
$2n^2 - n$. ................................................................. 76
4.10 Typical convergence behavior of a random $200 \times 200$ real symplec­
tic symmetric matrix. Note that a full sweep of $S_{4\alpha}$ and $S_{4\alpha\alpha}$ has
roughly half as many iterations as a full sweep of other strategies. 78
4.11 Relative change in off during first two sweeps .............................. 86
4.12 Detail of first sweep .............................................................. 86
4.13 Detail of second sweep ............................................................ 86
6.1 Typical convergence behavior of a $140 \times 140$ complex symplectic Hermitian matrix. Note that a full sweep of $\mathcal{CS}_4$ and $\mathcal{CS}_{4a}$ has roughly half as many iterations as a full sweep of other algorithms.
### Notation

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathbb{R}$</td>
<td>the real numbers</td>
</tr>
<tr>
<td>$\mathbb{C}$</td>
<td>the complex numbers</td>
</tr>
<tr>
<td>$\mathbb{K}$</td>
<td>the field $\mathbb{R}$ or $\mathbb{C}$</td>
</tr>
<tr>
<td>$\mathbb{R}^n$</td>
<td>vector space of real $n$ vectors</td>
</tr>
<tr>
<td>$\mathbb{C}^n$</td>
<td>vector space of complex $n$ vectors</td>
</tr>
<tr>
<td>$\mathbb{R}^{m \times n}$</td>
<td>vector space of real $m \times n$ matrices</td>
</tr>
<tr>
<td>$\mathbb{C}^{m \times n}$</td>
<td>vector space of complex $m \times n$ matrices</td>
</tr>
<tr>
<td>$I_n$</td>
<td>$n \times n$ identity matrix</td>
</tr>
<tr>
<td>$I$</td>
<td>identity matrix</td>
</tr>
<tr>
<td>0</td>
<td>zero scalar, vector, or matrix</td>
</tr>
<tr>
<td>$x = [x_1, x_2, \ldots, x_n]$</td>
<td>the row vector in $\mathbb{K}^n$</td>
</tr>
<tr>
<td>$x = [x_1, x_2, \ldots, x_n]^T$</td>
<td>the column vector in $\mathbb{K}^n$</td>
</tr>
<tr>
<td>$\Sigma$</td>
<td>diagonal matrix</td>
</tr>
<tr>
<td>$\text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$</td>
<td>diagonal matrix with diagonal entries $\sigma_1, \sigma_2, \ldots, \sigma_n$</td>
</tr>
<tr>
<td>$A^T$</td>
<td>transpose of matrix $A$</td>
</tr>
<tr>
<td>$A^*$</td>
<td>conjugate transpose of matrix $A$</td>
</tr>
<tr>
<td>$A^\star$</td>
<td>symplectic adjoint of matrix $A$</td>
</tr>
<tr>
<td>$A^{-1}$</td>
<td>inverse of the nonsingular matrix $A$</td>
</tr>
<tr>
<td>$\bar{A}$</td>
<td>complex conjugate of $A$</td>
</tr>
<tr>
<td>$A^{1/2}$</td>
<td>principal square root of $A$</td>
</tr>
<tr>
<td>$\text{rank}(A)$</td>
<td>rank of matrix $A$</td>
</tr>
<tr>
<td>$a_{ij}, A(i,j), (A)_{ij}$</td>
<td>the $(i,j)$ entry of $A$</td>
</tr>
<tr>
<td>$i = 1 : n$</td>
<td>$i = 1, 2, \ldots, n$</td>
</tr>
<tr>
<td>$A(i,:) , A(i,:) , (A)_{i,:}$</td>
<td>$i$th row of $A$</td>
</tr>
<tr>
<td>$A(:,j), A(:,j), (A)_{ :,j}$</td>
<td>$j$th column of $A$</td>
</tr>
<tr>
<td>$A(k,i:j)$</td>
<td>the entries $i,i+1,\ldots,j$ of the $k$th row of $A$</td>
</tr>
<tr>
<td>$A(i:j,k)$</td>
<td>the entries $i,i+1,\ldots,j$ of the $k$th column of $A$</td>
</tr>
<tr>
<td>$A(i:j,k:l)$</td>
<td>submatrix of $A$ formed by the intersection of rows $i$ to $j$ and columns $k$ to $l$</td>
</tr>
<tr>
<td>$|A|_F$</td>
<td>Frobenius norm of $A \in \mathbb{K}^{m \times n}$</td>
</tr>
<tr>
<td>$|x|, |x|_2$</td>
<td>Euclidean norm or length of vector $x \in \mathbb{K}^n$</td>
</tr>
<tr>
<td>$</td>
<td>x</td>
</tr>
<tr>
<td>$\bar{x}$</td>
<td>conjugate of scalar $x$</td>
</tr>
<tr>
<td>$\text{sign}(\tau)$</td>
<td>sign of real number $\tau$ ($\pm 1$)</td>
</tr>
<tr>
<td>$O(2n, \mathbb{R})$</td>
<td>the group of $2n \times 2n$ real orthogonal matrices</td>
</tr>
<tr>
<td>$U(2n)$</td>
<td>the group of $2n \times 2n$ unitary matrices</td>
</tr>
<tr>
<td>$Sp(2n, \mathbb{R})$</td>
<td>the group of $2n \times 2n$ real symplectic matrices</td>
</tr>
<tr>
<td>$Sp(2n, \mathbb{C})$</td>
<td>the group of $2n \times 2n$ complex symplectic matrices</td>
</tr>
</tbody>
</table>
$Sp^*(2n, \mathbb{C})$ the group of $2n \times 2n$ conjugate symplectic matrices

$SpO(2n, \mathbb{R}) = Sp(2n, \mathbb{R}) \cap O(2n, \mathbb{R})$

$SpU(2n, \mathbb{C}) = Sp(2n, \mathbb{C}) \cap U(2n)$

$Sp^*U(2n, \mathbb{C}) = Sp^*(2n, \mathbb{C}) \cap U(2n)$
Chapter 1
Introduction

One of the most useful matrix decompositions in the theory of matrices is the singular value decomposition, often abbreviated SVD. This decomposition was discovered independently by Beltrami (1873), Jordan (1874), and Sylvester (1889), and related work was done by Autonne (1915), Takagi (1925), Williamson (1935), Eckart and Young (1939), and others. For further historical discussion, see [18, page 134] and [36].

The following theorem shows that every matrix has an SVD.

Theorem 1.1 (Singular Value Decomposition). Let $A \in \mathbb{C}^{m \times n}$ and $d = \min\{m, n\}$. There is a matrix $\Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_d) \in \mathbb{C}^{m \times n}$, where $\sigma_i \geq 0$, $i = 1, 2, \ldots, d$, and unitary matrices $U \in \mathbb{C}^{m \times m}$ and $V \in \mathbb{C}^{n \times n}$ such that $A = U \Sigma V^*$. If $A \in \mathbb{R}^{m \times n}$, then $U$ and $V$ can be taken to be real orthogonal.

Proof. See, for example, [17, page 414], [18, page 144], [39, page 392]. □

The diagonal elements $\sigma_1, \sigma_2, \ldots, \sigma_d$ of $\Sigma$ are known as the singular values of $A$. The usual convention is to arrange the singular values of $A$ in decreasing order $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_d \geq 0$ on the diagonal, thus making $\Sigma$ unique. The number of positive singular values of $A$ is evidently equal to the rank of $A$. The columns of $U$ and $V$ are called left singular vectors and right singular vectors of $A$ respectively.

The SVD is an extremely useful tool in numerical linear algebra from which a variety of problems can be solved. It provides a wealth of information about the matrix, including orthogonal bases for the domain and range spaces, the 2-norm and Frobenius norm of the matrix, and a determination of the numerical rank of matrix. See for example [11] and references therein for the usefulness of the SVD in a wide variety of applications.

While every matrix has an SVD, such a decomposition is not unique. For example, we can always change sign of one column in $U$ and the corresponding column in $V$. Thus there exist other unitary matrices $U$ and $V$ satisfying $A = U \Sigma V^*$. In our work, the non-uniqueness of the SVD plays an important role which
we will discuss later.

Closely related to the SVD is the polar decomposition. It is analogous to the complex number representation $z = re^{i\theta}$, $r \geq 0$. The following theorem, taken from [15], is well known for this type of decomposition and can also be found in [17, page 412].

**Theorem 1.2 (Polar Decomposition).** Let $A \in \mathbb{C}^{m \times n}, m \geq n$. Then there exists a matrix $U \in \mathbb{C}^{m \times n}$ and a unique Hermitian positive semidefinite matrix $H \in \mathbb{C}^{n \times n}$ such that

$$A = UH, \quad U^*U = I_n.$$  

Furthermore, if $\text{rank}(A) = n$, then $U$ is uniquely determined and $H$ is positive definite, and can be expressed as $H = (A^*A)^{1/2}$.

As remarked in [16], the polar decomposition is an important theoretical and computational tool, and much is known about its approximation properties, its sensitivity to perturbations and its computation. See also [15] for further discussion including an application of the polar decomposition.

In this thesis we are interested in computing the SVD of two types of structured matrices — real symplectic matrices and complex symplectic matrices.

As mentioned earlier, any matrix has an SVD, all three factors of the decomposition are not determined uniquely. This leads to one of the crucial questions of our work. That is, if $A$ is symplectic, then do we have an SVD for which all three factors are also symplectic? Recently, Mackey, Mackey, Mehrmann [25] and Xu [42] have independently proved that every symplectic matrix has a symplectic SVD, thus answering the above question. The statement of the theorem about the symplectic SVD will be given in the Section 1.2. This theorem leads us to the main goal of our work, that is, finding a structure preserving algorithm for computing a symplectic SVD of a symplectic matrix.

In this thesis we present two ways of computing a structured SVD — one using the polar decomposition and another using one-sided Jacobi transformations. We organize our work as follows.

Sections 1.1 and 1.2 contain some basic definitions of symplectic groups and a theorem about the symplectic SVD, respectively.

In Chapter 2 structured tools for the real case used by our algorithms are discussed.

In Chapter 3 we present a condensed form for real symplectic symmetric matrices, the real symplectic bird form. This condensed matrix is also real symplectic symmetric, and it was described earlier by Bunse-Gerstner, Byers, and Mehrmann in [3]. Two different ways of achieving this reduction — using symplectic Householder transformations and using $4 \times 4$ symplectic Givens transformations — are discussed in detail.

Structure preserving algorithms for computing a real symplectic SVD using
the polar decomposition and using one-sided Jacobi transformations are discussed
in Chapters 4 and 5, respectively. Numerical experiments for all proposed algo­

rithms are also presented in these chapters.

In Chapter 6 extends the algorithms to complex symplectic matrices. Again
we use a polar decomposition and a one-sided Jacobi method, and present the
results of numerical experiments.

1.1 Symplectic Groups

We begin with a brief review of some definitions and properties of general bilinear
and sesquilinear forms taken from [28]. A more detailed discussion can be found,
for example, in [20], [22], or [35].

Let $\mathbb{K}$ denote either the field $\mathbb{R}$ or $\mathbb{C}$. Consider a map $(x, y) \mapsto \langle x, y \rangle$ from
$\mathbb{K}^n \times \mathbb{K}^n$ to $\mathbb{K}$. If such a map is linear in each argument, i.e.

$$
\langle \alpha_1 x_1 + \alpha_2 x_2, y \rangle = \alpha_1 \langle x_1, y \rangle + \alpha_2 \langle x_2, y \rangle,
$$

then it is called a bilinear form on $\mathbb{K}^n$. If $\mathbb{K} = \mathbb{C}$, and the map $(x, y) \mapsto \langle x, y \rangle$ is
conjugate linear in the first argument and linear in the second, i.e.

$$
\langle \alpha_1 x_1 + \alpha_2 x_2, y \rangle = \overline{\alpha_1} \langle x_1, y \rangle + \alpha_2 \langle x_2, y \rangle,
$$

then it is called a sesquilinear form.

Given a bilinear or sesquilinear form on $\mathbb{K}^n$, there exists a unique $M \in \mathbb{K}^{n \times n}$
such that for all $x, y \in \mathbb{K}^n$

$$
\langle x, y \rangle = \begin{cases} 
    x^T M y & \text{if the form is bilinear,} \\
    x^* M y & \text{if the form is sesquilinear.}
\end{cases}
$$

Here, the superscript $*$ is used for conjugate transpose. $M$ is called the matrix
associated with the form (with respect to the standard basis) and we will denote
$\langle x, y \rangle$ by $\langle x, y \rangle_M$ as needed.

A bilinear or sesquilinear form $\langle \cdot, \cdot \rangle_M$ is nondegenerate if the associated matrix
$M$ is nonsingular. Matrices that preserve the value of a nondegenerate real or
complex bilinear, or sesquilinear form $\langle x, y \rangle_M$ on $\mathbb{K}^n$ comprise the automorphism
group of the form, given by

$$
G = \{ A \in \mathbb{K}^{n \times n} : \langle Ax, Ay \rangle_M = \langle x, y \rangle_M, \forall x, y \in \mathbb{K}^n \}. 
$$

Note that the automorphism group $G$ always forms a multiplicative group, but it
is not a linear subspace.
For a fixed nondegenerate form $\langle \cdot, \cdot \rangle_M$ on $\mathbb{K}^n$, and for any matrix $A \in \mathbb{K}^{n \times n}$ there is a unique matrix $A^*$, the adjoint of $A$ with respect to $\langle \cdot, \cdot \rangle_M$, defined by

$$\langle Ax, y \rangle_M = \langle x, A^* y \rangle_M, \ \forall x, y \in \mathbb{K}^n.$$  

(1.2)

It can be shown by using (1.1) and (1.2) that the adjoint of an automorphism is its inverse:

$$A \in \mathcal{G} \iff A^* = A^{-1}.$$  

(1.3)

Thus, the adjoint can be used to characterize the automorphism group $\mathcal{G}$, that is,

$$\mathcal{G} = \{ A \in \mathbb{K}^{n \times n} : A^* = A^{-1} \}$$

An explicit formula for the adjoint is given by

$$A^* = \begin{cases} M^{-1} A^T M & \text{if the form is bilinear}, \\ M^{-1} A^* M & \text{if the form is sesquilinear}. \end{cases}$$  

(1.4)

Since the goal of this thesis is to find structure preserving algorithms for computing the symplectic singular value decomposition of symplectic matrices, we now turn our attention to the symplectic groups.

The symplectic groups arise in a variety of scientific applications, e.g., in discrete linear quadratic optimal control, the solution of discrete algebraic Riccati equations (see [7] and the references therein), and the eigenvalue problem for Hamiltonian matrices (see [1], [3]). These groups are the automorphism groups of the bilinear or sesquilinear form associated with the $2n \times 2n$ skew-symmetric matrix $J_{2n} = \left[ \begin{array}{cc} 0 & I_n \\ -I_n & 0 \end{array} \right]$. Note that $J_{2n}^{-1} = -J_{2n}$. Using (1.1) and (1.4) with $M = J_{2n}$, the symplectic groups can be defined as follows.

**Definition 1.3.** Let $\mathbb{K}$ denote either the field $\mathbb{R}$ or $\mathbb{C}$. $Sp(2n, \mathbb{K})$ is the automorphism group of the bilinear form defined by $\langle x, y \rangle = x^T J_{2n} y$, and this group can be expressed as:

$$Sp(2n, \mathbb{K}) = \{ A \in \mathbb{K}^{2n \times 2n} : -J_{2n} A^T J_{2n} = A^{-1} \}. \quad (1.5)$$

We will refer to $Sp(2n, \mathbb{R})$ as the real symplectic group and $Sp(2n, \mathbb{C})$ as the complex symplectic group. Matrices in these groups will be referred to as real symplectic matrices or complex symplectic matrices, respectively.

**Definition 1.4.** $Sp^*(2n, \mathbb{C})$ is the automorphism group of the sesquilinear form defined by $\langle x, y \rangle = x^* J_{2n} y$, and it can be expressed as:

$$Sp^*(2n, \mathbb{C}) = \{ A \in \mathbb{C}^{2n \times 2n} : -J_{2n} A^* J_{2n} = A^{-1} \}. \quad (1.6)$$

We will refer to $Sp^*(2n, \mathbb{C})$ as the conjugate symplectic group, and matrices in this group will be referred to as conjugate symplectic matrices.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
We will use the term *symplectic group* to refer to $Sp(2n, \mathbb{R})$ or $Sp(2n, \mathbb{C})$ or $Sp^*(2n, \mathbb{C})$, and the term *symplectic matrix* to refer to a matrix in any one of these three types of symplectic groups.

Thus,

$$A \in Sp(2n, \mathbb{K}) \Rightarrow A^* = -J_{2n}A^TJ_{2n} = A^{-1} \quad (1.7)$$
$$A \in Sp^*(2n, \mathbb{C}) \Rightarrow A^* = -J_{2n}A^*J_{2n} = A^{-1} \quad (1.8)$$

Moreover, both symplectic adjoints are involutory and commute with $\ast$, as is shown by the following calculation.

$$(A^*)^* = (-J_{2n}A^*J_{2n})^* = -J_{2n}(-J_{2n}A^*J_{2n})^*J_{2n} = -J_{2n}(-J_{2n}AJ_{2n})J_{2n} = A, \quad \forall A \in \mathbb{K}^{n \times n}, \quad (1.9)$$

and

$$(A^*)^* = -J_{2n}(A^*)^*J_{2n} = (-J_{2n}A^*J_{2n})^* = (A^*)^*, \quad \forall A \in \mathbb{K}^{n \times n}. \quad (1.10)$$

As remarked in [27], not all bilinear and sesquilinear forms satisfy (1.9) or (1.10).

A useful observation is that the symplectic adjoint preserves unitariness, that is, if $U$ is unitary, so is $U^\ast$. In fact, any adjoint satisfying (1.10) has this property, as is shown by the following calculation, taken from [25].

$$UU^* = I \implies (U^\ast)^*U^\ast = (U^\ast)^*U^\ast = (UU^\ast)^* = I.$$

Finally, we consider real diagonal matrices in the symplectic groups. Let $\Sigma = \text{diag}(\Sigma_1, \Sigma_2)$ where $\Sigma_1, \Sigma_2 \in \mathbb{R}^{n \times n}$ are diagonal matrices. As has been pointed out in [25], if $\Sigma \in Sp(2n, \mathbb{K})$ or $\Sigma \in Sp^*(2n, \mathbb{C})$, then from (1.7) and (1.8) we have that $\Sigma_2 = \Sigma_1^{-1}$. Therefore, real symplectic diagonal matrices have the form

$$\Sigma = \text{diag}(\Sigma_1, \Sigma_1^{-1}). \quad (1.11)$$

### 1.1.1 Symplectic Unitary Groups

A characterization of the symplectic orthogonal and symplectic unitary groups was given in [26]. We include those results here as these characterizations will be extensively used in our work.

From (1.5) it follows that a real orthogonal matrix is real symplectic if and only if it commutes with $J_{2n} = [0 I_n; -I_n 0]$. A direct computation shows that a
$2n \times 2n$ real matrix commutes with $J_{2n}$ if and only if it has the block structure $\begin{bmatrix} E & G \\ -G & E \end{bmatrix}$. So a real symplectic orthogonal group, which we denote by $SpO(2n, \mathbb{R})$, is given by

$$SpO(2n, \mathbb{R}) = \left\{ \begin{bmatrix} E & G \\ -G & E \end{bmatrix} \in O(2n, \mathbb{R}), \ E, G \in \mathbb{R}^{n \times n} \right\}, \quad (1.12)$$

where $O(2n, \mathbb{R})$ is the group of $2n \times 2n$ real orthogonal matrices.

Once again from (1.5), a $2n \times 2n$ unitary matrix $U$ is complex symplectic if and only if it satisfies the equation

$$J_{2n}U = UJ_{2n}. \quad (1.13)$$

We can show by using (1.13) that $U$ has the block structure $\begin{bmatrix} E & G \\ -G & E \end{bmatrix}$. So a complex symplectic unitary group, which we denote by $SpU(2n, \mathbb{C})$, is given by

$$SpU(2n, \mathbb{C}) = \left\{ \begin{bmatrix} E & G \\ -G & E \end{bmatrix} \in U(2n), \ E, G \in \mathbb{C}^{n \times n} \right\}, \quad (1.14)$$

where $U(2n)$ is the group of $2n \times 2n$ unitary matrices.

From (1.6) it follows that a unitary matrix is conjugate symplectic if and only if it commutes with $J_{2n}$. A direct computation shows that a $2n \times 2n$ complex matrix commutes with $J_{2n}$ if and only if it has the block structure $\begin{bmatrix} E & G \\ -G & E \end{bmatrix}$. So a conjugate symplectic unitary group, which we denote by $Sp^*U(2n, \mathbb{C})$, is given by

$$Sp^*U(2n, \mathbb{C}) = \left\{ \begin{bmatrix} E & G \\ -G & E \end{bmatrix} \in U(2n), \ E, G \in \mathbb{C}^{n \times n} \right\}, \quad (1.15)$$

where, as before, $U(2n)$ is the group of $2n \times 2n$ unitary matrices.

### 1.2 Symplectic Singular Value Decomposition

As remarked earlier, every symplectic matrix has a symplectic SVD. The following theorem taken from [25] and [42] ensures that for a symplectic matrix in a symplectic group, there is an SVD factorization that stays in the group.

**Theorem 1.5 (Symplectic Singular Value Decompositions).**

1. Let $A \in Sp(2n, \mathbb{R})$. Then $A$ has a real symplectic singular value decomposition, that is, there exist real symplectic orthogonal matrices $U$ and $V$, and a symplectic diagonal matrix $\Sigma = \text{diag}(\Sigma_1, \Sigma_1^{-1})$ such that $A = U\Sigma V^T$. Furthermore, $\Sigma_1 = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$ can be chosen to have $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 1$. 

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
2. Let \( A \in \text{Sp}(2n, \mathbb{C}) \). Then \( A \) has a complex symplectic singular value decomposition, that is, there exist unitary matrices \( U \) and \( V \) that are also complex symplectic, and a real symplectic diagonal matrix \( \Sigma = \text{diag}(\Sigma_1, \Sigma_1^{-1}) \) such that \( A = U\Sigma V^* \). Furthermore, \( \Sigma_1 = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n) \) can be chosen to have \( \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 1 \).

3. Let \( A \in \text{Sp}^*(2n, \mathbb{C}) \). Then \( A \) has a conjugate symplectic singular value decomposition, that is, there exist unitary matrices \( U \) and \( V \) that are also conjugate symplectic, and a real symplectic diagonal matrix \( \Sigma = \text{diag}(\Sigma_1, \Sigma_1^{-1}) \) such that \( A = U\Sigma V^* \). Furthermore, \( \Sigma_1 = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n) \) can be chosen to have \( \sigma_1 > \sigma_2 > \ldots > \sigma_n > 1 \).

**Proof.** See [25] or [42].

As has been pointed out at the beginning, the SVD is closely related to the polar decomposition. Given a polar decomposition \( A = UH \in \mathbb{C}^{n \times n} \), one can construct an SVD of \( A \) from a spectral decomposition of the Hermitian factor \( H \). That is,

\[
A = UH, \text{ where } U^*U = I, \ H^* = H \\
= U(Q\Sigma Q^*), \ \text{substituting } H = Q\Sigma Q^* \text{ with } Q^*Q = I \\
= (UQ)\Sigma Q^* 
\]

(1.16)
gives an SVD of \( A \).

Let \( A \) be in a symplectic group. Then \( A \) is invertible, and hence by Theorem 1.2 \( A \) has the polar decomposition

\[
A = UH, 
\]

where \( U \) is unitary and \( H \) is Hermitian positive definite. As was proved in [26], the factors \( U \) and \( H \) belong to the symplectic group. Moreover, [16] also gave structure preserving iterations for computing the symplectic unitary polar factor \( U \).

One way to achieve the main goal of this thesis, that is, finding a structure preserving algorithm for computing a symplectic SVD of a symplectic matrix \( A \) is to reduce our problem to finding structure preserving algorithms for computing a structured spectral decomposition of the Hermitian positive definite factor \( H \). The following corollary ensures that such a structured spectral decomposition exists.

**Corollary 1.6** (Spectral Decomposition of Positive Definite Symplectic Matrices).

a. Let \( A \) be a real symplectic symmetric positive definite matrix. Then there is a real symplectic orthogonal matrix \( Q \) and a real symplectic diagonal matrix \( \Sigma = \text{diag}(\Sigma_1, \Sigma_1^{-1}) \) such that \( A = Q\Sigma Q^T \). Furthermore, \( \Sigma_1 = \text{diag}(\lambda_1, \lambda_2, \cdots, \lambda_n) \) can be chosen to have \( \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 1 \).
b. Let $A$ be a complex symplectic Hermitian positive definite matrix. Then there is a complex symplectic unitary matrix $Q$ and a real symplectic diagonal matrix $\Sigma = \text{diag}(\Sigma_1, \Sigma_1^{-1})$ such that $A = Q\Sigma Q^*$. Furthermore, $\Sigma_1 = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ can be chosen to have $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 1$.

c. Let $A$ be a conjugate symplectic Hermitian positive definite matrix. Then there is a conjugate symplectic unitary matrix $Q$ and a real symplectic diagonal matrix $\Sigma = \text{diag}(\Sigma_1, \Sigma_1^{-1})$ such that $A = Q\Sigma Q^*$. Furthermore, $\Sigma_1 = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ can be chosen to have $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n > 1$.

Proof. We prove (a); the proof of (b) and (c) can be obtained by replacing an orthogonal matrix with a unitary matrix. Let $A$ be a real symplectic symmetric positive definite matrix. Then by Theorem 1.5, $A$ has a real symplectic singular value decomposition

$$A = U\Sigma V^T$$

(1.17)

where $U$ and $V$ are real symplectic orthogonal matrices and $\Sigma = (\Sigma_1, \Sigma_1^{-1})$ is a real symplectic diagonal matrix of singular values. Since $A$ is symplectic, $A$ is invertible, and all its singular values hence are positive. Furthermore, by Theorem 1.5, $\Sigma_1 = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$ can be chosen to have $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 1$.

The strategy of the proof is to use the polar decomposition to show that the matrices $U$ and $V$ are the same. We begin by noting that (1.17) can be written as

$$A = U\Sigma V^T = (UV^T)(V\Sigma V^T) = PH.$$  

(1.18)

Observe that $P := UV^T$ is real orthogonal and $H := V\Sigma V^T$ is real symmetric positive definite. Thus (1.18) displays a polar decomposition of $A$. But $A = IA$ gives another polar decomposition for $A$, since $A$ is symmetric positive definite.

Now we have two polar decompositions for $A$,

$$A = PH = IA.$$  

(1.19)

But the polar factors of $A$ are uniquely determined since $A$ is invertible. Thus, $P = UV^T = I$, i.e. $U = V$ and $H = V\Sigma V^T = A$. Setting $V = Q$, we have $A = Q\Sigma Q^T$ as desired. \qed
Chapter 2
Structured Tools

The primary goal of this chapter is to describe the tools used by our algorithms. To begin, we focus on structure preserving tools that can be used on real symplectic matrices that are also symmetric.

Since the algorithms we develop are Jacobi-like, we begin with a description of the basic facts about one of the oldest methods to solve the $n \times n$ symmetric eigenvalue problem — the Jacobi method.

2.1 The Classical Jacobi Method

Introduced in 1846 by C. G. J. Jacobi [19], this method transforms a real $n \times n$ symmetric matrix to diagonal form by a sequence of orthogonal similarity transformations. Each transformation, often called a Jacobi rotation, is just a plane rotation designed to annihilate one of the off-diagonal matrix elements.

The basic step in a Jacobi procedure involves choosing a principal $2 \times 2$ sub-matrix, which we denote simply by

$$
\begin{bmatrix}
a & z \\
z & b
\end{bmatrix}
$$

and a rotation angle $\theta$, so that

$$
\begin{bmatrix}
c & s \\ -s & c
\end{bmatrix}^T \begin{bmatrix}
a & z \\
z & b
\end{bmatrix} \begin{bmatrix}
c & s \\ -s & c
\end{bmatrix}, \text{ where } c = \cos \theta \text{ and } s = \sin \theta,
$$

(2.1)

is diagonal. Setting the off-diagonal element in the product (2.1) to zero gives

$$(a - b)cs + z(c^2 - s^2) = 0,$$

(2.2)

and hence, when $a \neq b$,

$$
\tan 2\theta = \frac{2z}{b - a}.
$$

(2.3)
Equation (2.3) constrains the angle \( \theta \) of a Jacobi rotation for an annihilating similarity. In general there are two choices for \( \theta \). A systematic description was given by Mackey [29], and is reproduced here:

- If \( z \neq 0 \) and \( a \neq b \), then using the identity \( \tan 2\theta = \frac{2\tan \theta}{1 - \tan^2 \theta} \) in (2.3) gives the quadratic equation

\[
\tan^2 \theta + \frac{b - a}{z} \tan \theta - 1 = 0. \tag{2.4}
\]

If \( \theta_1 < \theta_2 \) are the two solutions to (2.4) in the interval \( (-\frac{\pi}{2}, \frac{\pi}{2}) \), then

\[
\theta_2 = \theta_1 + \frac{\pi}{2},
\]

since from (2.4) we have \( \tan \theta_1 \tan \theta_2 = -1 \).

- If \( z \neq 0 \) and \( a = b \), then \( \theta = \pm \frac{\pi}{4} \). In this case define \( \theta_1 = -\frac{\pi}{4} \) and \( \theta_2 = \frac{\pi}{4} \).

- If \( z = 0 \) and \( a \neq b \), then \( \theta = 0, \pm \frac{\pi}{2} \). In this case define \( \theta_1 = 0 \) and \( \theta_2 = \frac{\pi}{2} \).

- If \( z = 0 \) and \( a = b \), then any \( \theta \) will give an annihilating similarity. In this case define \( \theta_1 = 0 \) and \( \theta_2 = \frac{\pi}{2} \).

The following definition taken from [29] defines the small angle and the large angle in terms of \( \theta_1 \) and \( \theta_2 \).

**Definition 2.1.** The small angle, \( \theta_{\text{small}} \), and the large angle, \( \theta_{\text{large}} \), are defined by

\[
\theta_{\text{small}} = \begin{cases} 
\theta_1 & \text{if } |\theta_1| \leq |\theta_2|, \\
\theta_2 & \text{otherwise.}
\end{cases}
\]

\[
\theta_{\text{large}} = \begin{cases} 
\theta_2 & \text{if } |\theta_1| \leq |\theta_2|, \\
\theta_1 & \text{otherwise.}
\end{cases}
\]

Note that with this definition \( \theta_{\text{small}} \in [-\frac{\pi}{4}, \frac{\pi}{4}] \).

Observe that whenever \( z \neq 0 \) and \( a \neq b \), from (2.3) we have

\[
\tan \theta_{\text{small}} = \frac{\text{sign}(\tau)}{|\tau| + \sqrt{\tau^2 + 1}}, \tag{2.5}
\]

where \( \tau \) is defined as

\[
\tau = \frac{b - a}{2z}.
\]
Moreover, choosing either $\theta = \theta_{\text{small}}$ or $\theta = \theta_{\text{large}}$ both $c = \cos \theta$ and $s = \sin \theta$ can be computed by using the identities $\tan^2 \theta + 1 = \sec^2 \theta$ and $\tan \theta = \frac{\sin \theta}{\cos \theta}$. That is, 

$$c = \frac{1}{\sqrt{t^2 + 1}} \quad \text{and} \quad s = tc, \quad \text{where} \quad t = \tan \theta. \quad (2.6)$$

The action of the similarity induced by $\theta_{\text{small}}$ and that induced by $\theta_{\text{large}}$ makes a difference in the order in which the new elements appear on the diagonal. The following computation taken from [29] shows that $\theta_{\text{small}}$ always preserves the order of the diagonal elements, whereas $\theta_{\text{large}}$ has the effect of reversing their order. Let $\tilde{a}$ and $\tilde{b}$ be the new value of $a$ and $b$, respectively. Then

$$\tilde{a} = a \cos^2 \theta + b \sin^2 \theta - 2z \cos \theta \sin \theta, \quad \tilde{b} = b \cos^2 \theta + a \sin^2 \theta + 2z \cos \theta \sin \theta.$$ 

Hence,

$$\tilde{a} - \tilde{b} = (a - b) \cos^2 \theta + (b - a) \sin^2 \theta - 4z \cos \theta \sin \theta,$$

$$= (a - b)(\cos^2 \theta - \sin^2 \theta) - 4z \cos \theta \sin \theta,$$

$$= (a - b) \cos 2\theta - 2z \sin 2\theta,$$

$$= (a - b) \cos 2\theta - (b - a) \tan 2\theta \sin 2\theta \quad \text{(using (2.3))},$$

$$= (a - b) \left( \cos 2\theta + \frac{\sin^2 2\theta}{\cos 2\theta} \right),$$

$$= \frac{1}{\cos 2\theta}(a - b).$$

If $-\pi/4 < \theta < \pi/4$, that is, when $\theta = \theta_{\text{small}}$, then $\cos 2\theta$ is positive, and hence $(\tilde{a} - \tilde{b})$ and $(a - b)$ will have the same signs. Conversely, $(\tilde{a} - \tilde{b})$ and $(a - b)$ will have the opposite signs, if $\theta = \theta_{\text{large}}$. This leads to the notion of the "sorting angle" [29, 30].

**Definition 2.2.** The sorting angle, $\theta_{\text{sort}}$, is defined by

$$\theta_{\text{sort}} = \begin{cases} 
\theta_{\text{large}} & \text{if } a < b, \\
\theta_{\text{small}} & \text{otherwise.}
\end{cases}$$

By using the sorting angle consistently, the diagonal elements move towards being in decreasing order as the definition suggests.

Now, if we denote the original $m \times m$ symmetric matrix by $A^{(1)}$, then the Jacobi method produces a sequence of orthogonally similar symmetric matrices $A^{(\ell)} = [a^{(\ell)}_{pq}]$, where

$$A^{(\ell + 1)} = R^T_{pq} A^{(\ell)} R_{pq}, \quad p \neq q, \quad (2.7)$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
and \( R_{pq} \) is a plane rotation of the form

\[
R_{pq} = \begin{bmatrix}
1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\
\vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\
0 & \cdots & c & \cdots & s & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
0 & \cdots & -s & \cdots & c & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & \cdots & 0 & \cdots & 1
\end{bmatrix}_{m \times m}
\] (2.8)

Here \( c = \cos \theta \) and \( s = \sin \theta \) for some \( \theta \). Notice that \( R_{pq} \) is obtained by embedding a \( 2 \times 2 \) rotation \([c\ s; -s\ c]\) as a principal submatrix into rows and columns \( p, q \) of an \( m \times m \) identity matrix. Furthermore, a similarity transformation via \( R_{pq} \) will affect only rows and columns \( p, q \) of \( A^{(\ell)} \).

Setting the off-diagonal element \( a_{pq}^{(\ell+1)} \) in (2.7) to zero yields

\[
a_{pq}^{(\ell+1)} = (a_{pp}^{(\ell)} - a_{qq}^{(\ell)})c^{(\ell)}s^{(\ell)} + a_{pq}^{(\ell)}(c^{(\ell)}s^{(\ell)} - s^{(\ell)}c^{(\ell)}) = 0,
\] (2.9)

and hence,

\[
\tan 2\theta^{(\ell)} = \frac{2a_{pq}^{(\ell)}}{a_{pp}^{(\ell)} - a_{qq}^{(\ell)}},
\] (2.10)

which is analogous to (2.3). Choosing either the small angle or the large angle leads to

\[
(c, s) = \left( \frac{1}{\sqrt{t^2 + 1}}, t \right), \text{ where } t = \tan \theta^{(\ell)}.
\] (2.11)

Since \( R_{pq} \) is orthogonal, the similarity transformation given by (2.7) preserves the Frobenius norm, that is,

\[
\|A^{(\ell+1)}\|_F = \|A^{(\ell)}\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{m} |a_{ij}^{(\ell)}|^2}.
\] (2.12)

Additional facts (see for example [29]) are true about the other off-diagonal elements. Consider row \( i \), where \( i \neq p, q \). Only two elements, the elements in columns \( p \) and \( q \), are affected in this row, and we find that

\[
(a_{ip}^{(\ell+1)})^2 + (a_{iq}^{(\ell+1)})^2 = (a_{ip}^{(\ell)})^2 + (a_{iq}^{(\ell)})^2, \quad i \neq p, q.
\] (2.13)

Similarly, considering column \( i \), we have

\[
(a_{pi}^{(\ell+1)})^2 + (a_{qi}^{(\ell+1)})^2 = (a_{pi}^{(\ell)})^2 + (a_{qi}^{(\ell)})^2, \quad i \neq p, q.
\] (2.14)
Equations (2.13) and (2.14) show that the affected elements change in a pairwise manner. We say that the pair $a_{ip}^{(\ell)}$ and $a_{jq}^{(\ell)}$ are coupled (or partners) during the transformation. Similarly, the pair $a_{pi}^{(\ell)}$ and $a_{qj}^{(\ell)}$ are coupled during the $\ell$th transformation.

Combining (2.12) - (2.14) and using the fact that elements not in rows $p$, $q$ and columns $p$, $q$ remain unchanged, we conclude that

$$(a_{pp}^{(\ell+1)})^2 + (a_{qq}^{(\ell+1)})^2 = (a_{pp}^{(\ell)})^2 + (a_{qq}^{(\ell)})^2 + 2(a_{pq}^{(\ell)})^2.$$ 

Thus, the “weight” of the annihilated elements $a_{pq}^{(\ell)}$ and $a_{qp}^{(\ell)}$ (which are equal by symmetry) has been moved onto the diagonal, so the weight of the off-diagonal part decreases. Hence, if we define

$$\text{off}(A) = \sqrt{\sum_{i=1}^{m} \sum_{j=1, j \neq i}^{m} a_{ij}^2},$$

then

$$(\text{off}(A^{(\ell+1)}))^2 = (\text{off}(A^{(\ell)}))^2 - 2(a_{pq}^{(\ell)})^2.$$ 

Therefore, the idea behind the Jacobi method is to move the “weight” of the annihilated elements onto the diagonal, while preserving the overall weight of the matrix.

The target element for annihilation, $a_{pq}^{(\ell)}$, is called the “pivot”, or the “pivot element”. So if $a_{pq}^{(\ell)}$ is the pivot at iteration $\ell$, then $a_{pq}^{(\ell+1)} = 0 = a_{qp}^{(\ell+1)}$. The non-pivot elements that are changed by the similarity are called the “affected” elements.

To complete the Jacobi method, a strategy for determining the rotation index pair $(p, q)$ in each step is needed. Jacobi suggests choosing $(p, q)$ so that $(a_{pq}^{(\ell)})^2$ is maximal in each step, thereby maximizing the reduction of off$(A)$. This strategy for choosing the pair $(p, q)$ is known as the Classical Jacobi Method. The classical Jacobi method is a reasonable strategy when computing by hand on a small matrix because it is a simple matter to identify the optimal $(p, q)$, but the hard part is the update. However, for a larger matrix, searching for the optimal $(p, q)$ is expensive: it requires $O(n^2)$ comparisons at each iteration, whereas performing the similarity transformation by a plane rotation requires only $O(n)$ arithmetic operations [11].

Different ways of choosing pairs $(p, q)$ have been considered. For example, the strategy called the cyclic Jacobi method chooses the rotation index pairs in strict order. In the cyclic-by-row method the pairs are taken in the order

$$(1, 2), (1, 3), \ldots, (1, m), (2, 3), \ldots (2, m), (3, 4), \ldots, (m-1, m).$$

Since this strategy does not require off-diagonal search, it is considerably faster than the classical Jacobi method.
In the classical Jacobi method, the pivot in step $\ell$ is the largest off-diagonal entry $(p, q)$, and

$$(\text{off}(A^{(\ell)}))^2 \leq \frac{n(n - 1)}{2} (a_{pq}^{(\ell)} + a_{qp}^{(\ell)})^2,$$

from (2.16) it follows that

$$(\text{off}(A^{(\ell+1)}))^2 \leq \left[1 - \frac{2}{n(n - 1)}\right] (\text{off}(A^{(\ell)}))^2.$$

This implies that the rate of convergence of the classical Jacobi method is linear. However, the asymptotic convergence rate of this method is quadratic, see for example [11] and references therein.

Establishing the convergence of cyclic Jacobi methods has attracted the attention of many researchers. For example, in 1960, Forsythe and Henrici [9] showed that the row-cyclic method converges if the angle is suitable restricted. In 1960, Henrici [13] proved that for any $n \times n$ matrix with $n$ distinct eigenvalues the row-cyclic method converges quadratically when the angle used at each iteration is confined to $[-\pi, \pi]$. In his 1991 thesis, Mascarenhas [30] presents a direct proof of convergence of quasi-cyclic plane Jacobi methods for symmetric matrices. In his proof, these are two main steps, first finding sufficient conditions under which the sequence of diagonals of the Jacobi-iteration converges to a point in $\mathbb{R}^n$ and then showing that the off-diagonal norm converges to zero. For further discussion about the convergence of the classical Jacobi method and cyclic Jacobi method see, for example, [12], [32], [40], [41].

### 2.2 Real Symplectic Plane Rotations

We now turn our attention to describing the tools used by our algorithms. As we noted earlier, we are searching for tools that can be used on a real symplectic symmetric matrix. Preserving the real symplectic symmetric structure of a given matrix restricts us to real symplectic orthogonal similarities, so desired tools are limited to the matrices from $SpO(2n, \mathbb{R})$, as given in (1.12).

Plane rotations as specified in (2.8) are clearly real orthogonal but not all of them are real symplectic. To be symplectic, such rotations must have block structure as described in (1.12); this implies that $m = 2n$ and $q = n + p$ in (2.8). Hence a real $2n \times 2n$ symplectic plane rotation is a matrix of the form
where \( c = \cos \theta \) and \( s = \sin \theta \) for some \( \theta \). The remaining entries of \( R_{p,n+p} \) correspond to those in the identity matrix \( I_{2n} \).

We now restrict our attention to studying the effect on different parts of a symmetric matrix \( A \in \mathbb{R}^{2n \times 2n} \), when it is acted on by a similarity transformation \( Q = R_{p,n+p} \) as in (2.17). The goal of the similarity transformation

\[
\hat{A} = Q^T AQ
\]

is, as always, to zero out an off-diagonal entry of \( A \), while preserving both the symplectic and symmetric structure of \( A \). There are two ways in which a symplectic plane rotation can achieve this goal:

1. By a Givens-like action: the entry to be zeroed out is affected either by \( Q^T \) from the left or by \( Q \) from the right, but not by both.

2. By a Jacobi-like action: the entry to be zeroed out is affected by both \( Q^T \) from the left or by \( Q \) from the right. This is what happens in the original Jacobi method for symmetric matrices.

Let \( A_{i,:} \) and \( A_{:,j} \) denote the \( i^{th} \) row and \( j^{th} \) column of \( A \), respectively. The similarity in (2.18) can be viewed as pre-multiplication by \( Q^T \) and post-multiplication by \( Q \) on the matrix \( A \).

Since \( Q \) is of the form as in (2.17), pre-multiplication by \( Q^T \) changes only rows \( p \) and \( n+p \), and we find that

\[
(Q^T A)_{p,:} = cA_{p,:} - sA_{n+p,:} \quad (2.19)
\]

\[
(Q^T A)_{n+p,:} = sA_{p,:} + cA_{n+p,:} \quad (2.20)
\]

As before in this case \( a_{pi} \) and \( a_{n+p,i} \) are partners during the transformation — the quantities

\[
a_{pi}^2 + a_{n+p,i}^2, \quad i = 1, 2, \ldots, 2n,
\]

are kept invariant. That is,

\[
(Q^T A)_{pi}^2 + (Q^T A)_{n+p,i}^2 = a_{pi}^2 + a_{n+p,i}^2, \quad i = 1, 2, \ldots, 2n. \quad (2.21)
\]
Similarly, post-multiplication by $Q$ changes only columns $p$ and $n+p$ and we have
\begin{align}
(AQ)_{i,p} &= cA_{i,p} - sA_{i,n+p} \\
(AQ)_{i,n+p} &= sA_{i,p} + cA_{i,n+p}
\end{align}
(2.22)
(2.23)
This transformation also pairs the elements $a_{ip}$ and $a_{i,n+p}$, and keeps the quantities
\[ a_{ip}^2 + a_{i,n+p}^2, \quad i = 1, 2, \ldots, 2n, \]
invariant. That is, we have
\[ (AQ)_{i,p}^2 + (AQ)_{i,n+p}^2 = a_{ip}^2 + a_{i,n+p}^2, \quad i = 1, 2, \ldots, 2n. \]
(2.24)

### 2.2.1 Givens-like Action

We are now interested in the zeroing action on $A$ either by $Q^T$ or $Q$, but not by both. We will refer to the zeroing action by pre-multiplication $Q^T A$ as a left Givens-like action or left action for short. Similarly, we will refer to the zeroing action by post-multiplication $A Q$ as a right Givens-like action or right action for short. Note that the matrix $Q$ in these cases is sometimes referred to as a real symplectic Givens rotation [37].

**Left Action**

In this case we want a new zero to appear in the product $Q^T A$, and further, we desire that this zero not be disturbed when the similarity is completed by post-multiplying by $Q$. The element to be targeted for annihilation should therefore be in one of the affected rows, but not in either of the affected columns. Thus, the pivot element is either $a_{pr}$ or $a_{n+p,r}$ where $r \notin \{p, n+p\}$. Without loss of generality, we may assume that $a_{pr} = 0$. Figure 2.1 shows the action $Q^T A$. Here the pivot element is marked by $\bigstar$; the entry represented by $\ast$ is its partner during the transformation. The entries in positions $(p, i)$ and $(n+p, i)$, $i = 1, \ldots, 2n$, are partners during the transformation. A typical pair of partners is marked by $\times$.

To annihilate the off-diagonal element $(p, r)$ of $Q^T A$, by using (2.19) we have that $c, s$ must satisfy
\[ cc_{pr} - s a_{n+p,r} = 0. \]
(2.25)
If $a_{pr} = 0$, then we just set $(c, s) = (1, 0)$, and $Q$ is just the $2n \times 2n$ identity matrix. Otherwise define
\[ \omega = \sqrt{a_{pr}^2 + a_{n+p,r}^2}. \]
(2.26)
Solving (2.25) one obtains
\[ c = \frac{a_{n+p,r}}{\omega} \quad \text{and} \quad s = \frac{a_{pr}}{\omega}. \]
(2.27)
Moreover, the negatives of $\omega$, $c$, and $s$ also satisfy (2.25). Hence, $\omega$, $c$, and $s$ are not determined uniquely. For more discussion on the choices of $\omega$, $c$, and $s$, see, for example [2].

Now consider column $r$ of $Q^T A$. Only two elements of column $r$, those in row $p$ and row $n + p$, are changed, and one of these is zeroed out. Hence, using the fact that the weight of the pair $a_{pr}$ and $a_{n+p,r}$ is unchanged as in (2.21) we have

$$a_{pr}^2 + a_{n+p,r}^2 = (Q^T A)^2_{pr} + (Q^T A)^2_{n+p,r} = (Q^T A)_{n+p,r},$$

that is, the left action moves the entire weight of the pivot element (marked $\circ$) to its partner (marked $\ast$).

To preserve the symmetric structure of $A$, the similarity $Q^T AQ$ is needed. Right multiplication of $Q^T A$ by $Q$ affects only columns $p$ and $n + p$ of $Q^T A$. So the zero introduced in column $r$, $r \notin \{p, n + p\}$, is preserved and one additional zero is forced, i.e. $(Q^T AQ)_{pr} = (Q^T AQ)_{r p} = 0$ by symmetry.

Pseudo-code for the computation of $Q$ is given below. A semi-colon is used to separate two commands appearing on the same line.

**function:** $Q = \text{Givens.left}(A, p, r, \text{tol})$

Given a real symmetric matrix $A \in \mathbb{R}^{2n \times 2n}$ and integers $p$ and $r$ that satisfy $1 \leq p \leq n$, $1 \leq r \leq 2n$, $r \notin \{p, n + p\}$, and $\text{tol} > 0$, this algorithm computes a real symplectic orthogonal transformation $Q \in \mathbb{R}^{2n \times 2n}$ as in (2.17) such that if $\hat{A} = Q^T AQ$, then $\hat{a}_{pr} = \hat{a}_{r p} = 0$.

if $|a_{pr}| > \text{tol}$ then

$$\omega = \sqrt{a_{pr}^2 + a_{n+p,r}^2}$$

$$c = \frac{a_{n+p,r}}{\omega}; \quad s = \frac{a_{pr}}{\omega}$$

else

$$c = 1; \quad s = 0$$

17

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
endif

\[ Q = I_{2n}; \quad Q(p, p) = c; \quad Q(p, n + p) = s; \quad Q(n + p, p) = -s; \quad Q(n + p, n + p) = c \]

Note that an algorithm for setting \( \hat{a}_{n+p,r} = \hat{a}_{r,n+p} = 0 \) can be obtained by replacing \( p \) and \( n + p \) with \( n + p \) and \( p \), respectively in function Givens.left.

**Right Action**

In this case, our aim is to zero an off-diagonal entry in the product \( AQ \) such that this introduced zero is preserved when the similarity is performed by pre-multiplying by \( Q^T \). Thus, the pivot element should be in one of the affected columns but not in the affected rows. Either \( a_{rp} \) or \( a_{r,n+p} \) where \( r \notin \{p, n + p\} \) can be the pivot element. Without loss of generality, we may assume that \( \hat{a}_{r,n+p} = 0 \).

Figure 2.2 shows this right action. Here the pivot element is denoted by \( \diamond \); its partner is marked by \( * \). The entries in positions \((i, p)\) and \((i, n+p)\), \(i = 1, 2, \ldots, 2n\), are coupled during transformation, with a typical coupled pair represented by \( x \).

\[
\begin{bmatrix}
p & n+p \\
\vdots & \vdots \\
i & \times & \times \\
\vdots & \vdots & \vdots \\
r & * & \diamond \\
\vdots & \vdots & \vdots \\
\end{bmatrix}
\begin{bmatrix}
p & n+p \\
\vdots & \vdots \\
i & \times & \times \\
\vdots & \vdots & \vdots \\
r & * & 0 \\
\vdots & \vdots & \vdots \\
\end{bmatrix}
\]

\[
A \\
AQ
\]

Figure 2.2: Right action: columns \( p, n + p \) changed

From (2.23) we see that setting the off-diagonal element \((r, n + p)\) of \( AQ \) to zero means that \( c, s \) must satisfy

\[ s a_{rp} + c a_{r,n+p} = 0. \]  

(2.28)

This equation is analogous to (2.25), so \((c, s)\) can be found in exactly the same way. That is, if \( a_{r,n+p} = 0 \), then we set \((c, s) = (1, 0)\), and hence \( Q \) is just \( I_{2n} \). Otherwise define

\[ \omega = \sqrt{a_{rp}^2 + a_{r,n+p}^2}. \]  

(2.29)

Then (2.28) yields

\[ c = \frac{a_{rp}}{\omega} \quad \text{and} \quad s = -\frac{a_{r,n+p}}{\omega}. \]  

(2.30)
As mentioned before, the negatives of $\omega$, $c$, and $s$ also satisfy (2.28), that is, $\omega$, $c$, and $s$ are not unique.

To investigate more about the affected off-diagonal elements, we begin with considering the row $r$ of $AQ$. Only two elements of row $r$, those in column $p$ and column $n+p$, of $QA$ are changed and one of these was made zero. Using (2.24) one obtains
\[
 a_{rp}^2 + a_{r,n+p}^2 = (AQ)_{rp}^2 + (AQ)_{r,n+p}^2 = (AQ)_{rp}^2 ,
\]
i.e., all the weight of the pivot element (denoted by $\circ$) has moved into its partner (denoted by $\ast$) by the right action.

To complete the similarity, we must premultiply $AQ$ by $Q^T$ to get $QTAQ$. This multiplication will effect only row $p$ and row $n+p$ of $AQ$; it will not destroy the introduced zero in row $r$, since $r \notin \{p, n+p\}$ and it will force one additional zero, i.e. $(QTAQ)_{r,n+p} = (QTAQ)_{n+p,r} = 0$, by symmetry.

The computation of $Q$ is given in the following.

\textbf{function:} $Q =$ Givens.right($A, r, n+p, \text{tol}$)

Given a real symmetric matrix $A \in \mathbb{R}^{2n \times 2n}$ and integers $r$ and $n+p$ that satisfy $1 \leq r \leq 2n$, $1 \leq p \leq n$, $r \notin \{p, n+p\}$, and $\text{tol} > 0$, this algorithm computes a real symplectic orthogonal transformation $Q \in \mathbb{R}^{2n \times 2n}$ as in (2.17) such that if $\hat{A} = Q^T AQ$, then $\hat{a}_{r,n+p} = \hat{a}_{n+p,r} = 0$.

\begin{verbatim}
if |a_{r,n+p}| > tol then
  $\omega = \sqrt{a_{rp}^2 + a_{r,n+p}^2}$
  $c = \frac{a_{rp}}{\omega}$; $s = \frac{-a_{r,n+p}}{\omega}$
else
  $c = 1$; $s = 0$
endif
$Q = I_{2n}$; $Q(p,p) = c$; $Q(p,n+p) = s$; $Q(n+p,p) = -s$; $Q(n+p,n+p) = c$
\end{verbatim}

Note that an algorithm for setting $\hat{a}_{rp} = \hat{a}_{pr} = 0$ can be obtained by replacing $p$ and $n+p$ with $n+p$ and $p$, respectively in function Givens.right.

\subsection{2.2.2 Jacobi-like Action}

Now we consider the case when the off-diagonal entry that is targeted for annihilation is $a_{n+p,n+p}$. Since it is in row $p$, it is changed by $Q^T$ acting from the left; since it is in column $n+p$, it is also changed by $Q$ acting from the right. Because symmetry of $A$ is preserved, $a_{n+p,n+p}$ is also zeroed out. Setting the elements $(p, n+p)$ and $(n+p, p)$ of $\hat{A}$ to zero yields
\[
\hat{a}_{p,n+p} = (a_{p,n+p} - a_{n+p,n+p})c + a_{p,n+p}(c^2 - s^2) = 0.
\]
This equation is the same as (2.9) when $q = n + p$, and hence $(c, s)$ can be obtained as in (2.11).

Note that, as discussed in Section 2.1, the weight of the entries $a_{p,n+p}$ and $a_{n+p,p}$ moves onto the diagonal entries $a_{pp}$ and $a_{n+p,n+p}$, and $(\text{off}(A))^2$ is reduced by $2a^2_{p,n+p}$, that is

$$(\text{off}(\hat{A}))^2 = (\text{off}(A))^2 - 2a^2_{p,n+p}. \quad (2.31)$$

By contrast, when Givens-like action is used to zero pivot elements, in general the weight of pivots does not move to diagonal. Thus $(\text{off}(A))^2$ does not, in general, decrease.

The computation of $Q$ is given in the following. Here, text following "%%" denotes comments.

**function:** $Q = \text{Rotation}(A, p, \text{sort}, \text{tol})$

Given a real symmetric matrix $A \in \mathbb{R}^{2n \times 2n}$, an integer $p$ that satisfies $1 \leq p \leq n$, a Boolean parameter "sort", and $\text{tol} > 0$, this algorithm computes a symplectic plane rotation $Q \in \mathbb{R}^{2n \times 2n}$ as in (2.17) such that if $\hat{A} = Q^T A Q$, then $\hat{a}_{p,n+p} = \hat{a}_{n+p,p} = 0$. If $\text{sort} = 0$, then the order of the diagonal elements $(p, p)$ and $(n + p, n + p)$ is preserved, that is, $\text{sign}(a_{pp} - a_{n+p,n+p}) = \text{sign}(\hat{a}_{pp} - \hat{a}_{n+p,n+p})$. If $\text{sort}$ is nonzero, then the sorting angle is used for ordering the diagonal elements $(p, p)$ and $(n + p, n + p)$ of $\hat{A}$ in decreasing order.

if $|a_{p,n+p}| > \text{tol}$ then
  \[\tau = \frac{a_{n+p,n+p} - a_{pp}}{2a_{p,n+p}}\]
  \[t = \frac{\text{sign}(\tau)}{|\tau| + \sqrt{\tau^2 + 1}}\] % as in (2.5)
  if ($\text{sort} \neq 0$) and ($a_{pp} < a_{n+p,n+p}$) then
    $t = -\frac{1}{t}$ % use large angle
  endif
  \[c = \frac{1}{\sqrt{t^2 + 1}}; \quad s = tc\]
else
  $c = 1; \quad s = 0$
endif
\[Q = I_{2n}; \quad Q(p, p) = c; \quad Q(p, n + p) = s; \quad Q(n + p, p) = -s; \quad Q(n + p, n + p) = c\]
2.3 Real Symplectic Double Rotations

Let $R = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}$ be a real $2 \times 2$ rotation. A coupled pair of rotations $R$ can be embedded as principal submatrices in rows and columns $p, q, n + p, n + q$ of $I_{2n}$, where $1 \leq p < q \leq n$, to obtain real symplectic double rotations.

Such rotations are an important tool used by our algorithms. Depending on the action desired, there are two ways to construct real symplectic double rotations. We now study the effect of these tools on a symmetric matrix $A \in \mathbb{R}^{2n \times 2n}$.

2.3.1 Direct Sum Embedding

As illustrated in (2.32), one copy of $R$ is embedded into rows and columns $p, q$, while the other copy is embedded into rows and columns $n + p, n + q$. The resulting matrix $Q$ is clearly orthogonal with block structure as described in (1.12). Thus $Q \in \text{SpO}(2n, \mathbb{R})$. This transformation was introduced by Paige and Van Loan in [33].

\[
Q = \begin{bmatrix} R & 0 \\ 0 & R \end{bmatrix}
\]

In order to keep the notation concise, the $n \times n$ plane rotation obtained by embedding the $2 \times 2$ rotation $R$ into the identity matrix has also been denoted by $R$. The aim of the similarity transformation

\[
\hat{A} = Q^T AQ,
\]

is to set one of the off-diagonal elements, either $\hat{a}_{pq}$ or $\hat{a}_{n+p,n+q}$ in (2.33) to zero, while preserving the structure of $A$. Without loss of generality, we may assume that $\hat{a}_{pq} = 0$. Thus the rotation in the upper diagonal block is the one that makes the $(p, q)$ entry zero. We will refer to the rotation used to annihilate a pivot element as the active rotation.

Let $A$ have block structure given by $A = \begin{bmatrix} E & F \\ F^T & H \end{bmatrix}$, where $E, H \in \mathbb{R}^{n \times n}$ are symmetric. Thus

\[
\hat{A} = Q^T AQ = \begin{bmatrix} R & 0 \\ 0 & R \end{bmatrix}^T \begin{bmatrix} E & F \\ F^T & H \end{bmatrix} \begin{bmatrix} R & 0 \\ 0 & R \end{bmatrix} = \begin{bmatrix} R^T ER & R^T FR \\ R^T F^T R & R^T HR \end{bmatrix},
\]

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
that is, each of the $n \times n$ blocks of $A$ simultaneously undergoes an orthogonal similarity by the same $n \times n$ plane rotation $R$. In particular, the Frobenius norm of all four $n \times n$ blocks of $A$ is individually preserved.

The element $(p, q)$ of $E$ is made zero by a Jacobi-like action, and hence the analysis of Section 2.1 tells us that $(c, s)$ can be obtained as in (2.11) and the weight of the off-diagonal part of $R^T E R$ decreases, that is,

$$\text{off}(R^T E R)^2 = \text{off}(E)^2 - 2a_{pq}^2.$$ 

The lower diagonal block $H$ is acted on by the inactive rotation, that is, none of the elements of $H$ are targeted for explicit annihilation. As before, when $i \geq n + 1$ and $i \notin \{n + p, n + q\}$, only two elements from the $i$th row, those in column $n + p$ and $n + q$, are affected, and we find that

$$(\tilde{a}_{i,n+p})^2 + (\tilde{a}_{i,n+q})^2 = (a_{i,n+p})^2 + (a_{i,n+q})^2, \quad i \geq n + 1, i \neq n + p, n + q, \quad (2.35)$$

that is, the weight of the pair $a_{i,n+p}$ and $a_{i,n+q}$ is invariant.

Similarly, considering column $i$, we obtain

$$(\tilde{a}_{n+p,i})^2 + (\tilde{a}_{n+q,i})^2 = (a_{n+p,i})^2 + (a_{n+q,i})^2, \quad i \geq n + 1, i \neq n + p, n + q, \quad (2.36)$$

that is, the weight of the pair $a_{n+p,i}$ and $a_{n+q,i}$ is also invariant.

Since the similarity $R^T H R$ preserves the Frobenius norm, combining (2.35)–(2.36) and using the fact that the elements not in rows $n + p, n + q$ and columns $n + p, n + q$ remain unchanged, we can conclude that

$$(\tilde{a}_{n+p,n+p})^2 + (\tilde{a}_{n+q,n+q})^2 + 2(\tilde{a}_{n+p,n+q})^2 = (a_{n+p,n+p})^2 + (a_{n+q,n+q})^2 + 2(a_{n+p,n+q})^2.$$ 

This implies that there is liable to be some “leakage” of the weight of the diagonal entries $a_{n+p,n+p}$ and $a_{n+q,n+q}$ into the off-diagonal entries $\tilde{a}_{n+p,n+q}$ and $\tilde{a}_{n+q,n+p}$. Consequently, after the similarity transformation, the weight of the off-diagonal part of $R^T H R$ could possibly increase.

If the weight of the off-diagonal part of $R^T H R$ increases by more than weight of the off-diagonal part of $R^T E R$ decreases, then the overall off-diagonal weight of $A$ increases as well. That this does happen in practice is confirmed by numerical experiments (see Table 4.18 in Section 4.4.1).

We expect that when we use a transformation of the form as in (2.32) in an algorithm, the norm of the off-diagonal could, in theory, oscillate. This suggests that a proof of convergence of such an algorithm cannot rely on the usual strategy of showing that the off-diagonal norm decreases with every iteration.

The computation of $Q$ is given in the following.

**function:** $Q = \text{Double}(A, p, q, \text{sort}, \text{tol})$

*Given a real symmetric matrix $A \in \mathbb{R}^{2n \times 2n}$, integers $p$ and $q$ that satisfy*
1 ≤ p < q ≤ n, a Boolean parameter "sort", and tol > 0, this algorithm computes a real symplectic orthogonal transformation \( Q \in \mathbb{R}^{2n \times 2n} \) as in (2.32) such that if \( \hat{A} = Q^T A Q \), then \( \hat{a}_{pq} = \hat{a}_{qp} = 0 \). If sort = 0, then the order of the diagonal elements \((p,p)\) and \((q,q)\) is preserved, that is, \( \text{sign}(a_{pp} - a_{qq}) = \text{sign}(a_{pp} - a_{qq}) \). If sort is nonzero, then the sorting angle is used for ordering the diagonal elements \((p,p)\) and \((q,q)\) of \( \hat{A} \) in decreasing order.

If \( |a_{pq}| > \text{tol} \) then

\[
\tau = \frac{a_{qp} - a_{pp}}{2a_{pq}} \quad \text{sign } \tau \quad \text{compute small angle}
\]

\[
t = \frac{\tau}{|\tau| + \sqrt{\tau^2 + 1}} \quad \text{as in (2.5)}
\]

if \((\text{sort} \neq 0 \text{ and } a_{pp} < a_{qq})\) then

\[
t = -\frac{1}{t} \quad \text{use large angle}
\]

else

\[
c = \frac{1}{\sqrt{t^2 + 1}} \quad s = tc
\]

end if

\[
Q = I_{2n}; \quad Q(p,p) = c; \quad Q(p,q) = s; \quad Q(q,p) = -s; \quad Q(q,q) = c;
\]

\[
Q(n+p,n+p) = c; \quad Q(n+p,n+q) = s; \quad Q(n+q,n+p) = -s; \quad Q(n+q,n+q) = c
\]

Note that an algorithm for setting \( \hat{a}_{n+p,n+q} = \hat{a}_{n+q,n+p} = 0 \), can be obtained by replacing \( p \) and \( q \) with \( n + p \) and \( n + q \), respectively in function Double.

**2.3.2 Concentric Embedding**

As shown in (2.37), one copy of the \( 2 \times 2 \) rotation \( R \) is embedded in rows and columns \( p, n + q \), and the second copy of \( R \) in rows and columns \( q, n + p \), where \( 1 \leq p < q \leq n \). It is easy to see that \( Q \) is orthogonal, and has block structure as described in (1.12). So, \( Q \in SpO(2n, \mathbb{R}) \). This transformation, described by Mackey, Mackey, and Tisseur in [26], does not seem to be as well-known as the direct sum embedding.

We now consider the effect of a similarity transformation on a symmetric matrix \( A \in \mathbb{R}^{2n \times 2n} \) by \( Q \) as in (2.37), \( \hat{A} = [\hat{a}_{ij}] = Q^T A Q \). The aim of this similarity is to set one of the off-diagonal elements, either the \((p, n+q)\) or \((q, n+p)\), of \( \hat{A} \) to zero. Without loss of generality, we may assume that \( \hat{a}_{p,n+q} = 0 \).

To understand the action of the similarity, we start by writing the symplectic matrix \( Q \) as the (commuting) product of two non-symplectic plane rotations, that is, \( Q = R_1 R_2 \) \((= R_2 R_1)\) where \( R_1 \) and \( R_2 \) are of the form as in (2.38) and (2.39),

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
respectively.

\[
Q = \begin{array}{cccc}
p & q & n+p & n+q \\
p & c & s & -s \\
q & c & s & -c \\
n+p & -s & c & -s \\
n+q & -c & s & c \\
\end{array} \tag{2.37}
\]

\[
R_1 = \begin{bmatrix}
p & n+q \\
pc & s \\
n+q & -s & c \\
\end{bmatrix}_{2n \times 2n}, \quad c = \cos \theta, \; s = \sin \theta, \tag{2.38}
\]

\[
R_2 = \begin{bmatrix}
q & n+p \\
c & s \\
n+p & -s & c \\
\end{bmatrix}_{2n \times 2n}, \quad c = \cos \theta, \; s = \sin \theta. \tag{2.39}
\]

Note that the remaining entries of \( R_1 \) and \( R_2 \) correspond to those in \( I_{2n} \). We will refer to \( R_1 \) as the outer rotation, and \( R_2 \) as the inner rotation. Thus, the similarity by \( Q \) on \( A \) can be written as

\[
\hat{A} = Q^T AQ = (R_1 R_2)^T A (R_1 R_2) = R_2^T (R_1^T A R_1) R_2. \tag{2.40}
\]

That is, this action can be viewed in 2 stages. First, a similarity is performed on \( A \) by using the outer rotation \( R_1 \), \( A^{(1)} = [a_{ij}^{(1)}] = R_1^T A R_1 \). Second, we use the inner rotation \( R_2 \) to perform a similarity on \( A^{(1)} \), \( A^{(2)} = [a_{ij}^{(2)}] = R_2^T A^{(1)} R_2 \). Observe that \( A^{(2)} \) is just \( \hat{A} \).

Let us now examine the affect of each similarity. Figure 2.3 shows the action of the first similarity, \( A^{(1)} = R_1^T A R_1 \). This similarity is performed by the outer active rotation that introduces a zero in position \((p, n+q)\) by a Jacobi-like action. Note that by symmetry the \((n+q, p)\) entry will also be zeroed out. This similarity...
Figure 2.3: Action by outer rotation $R_1$ on $A$

affects only rows and columns $p$ and $n + q$ of $A$ (marked $\cdot$). Here the entries denoted by $*$ carry all weight of the pivot elements.

The angle $\theta$ can be determined by using (2.9) when $q$ is replaced with $n + q$, that is,
\begin{equation}
(a_{p,n+q}^{(1)}) = a_{n+q,p}^{(1)} = (a_{pp} - a_{n+q,n+q}) \cos + a_{p,n+q}(c^2 - s^2) = 0,
\end{equation}
and hence $(c,s)$ can be obtained as in (2.11). Once again we get
\[(\text{off}(A^{(1)}))^2 = (\text{off}(A))^2 - 2a_{p,n+q}^2,
\]
that is, there is a reduction in the off-diagonal norm.

Next, we investigate the off-diagonal elements affected by the inner rotation $R_2$, shown in Figure 2.4. The similarity $A^{(2)} = R_2^T A^{(1)} R_2$ will affect only rows and columns $q$, $n + p$ (marked $\cdot \cdot \cdot$). This implies that the introduced zeroes in positions $(p,n+q)$ and $(n+q,p)$ by the first similarity are preserved. Since the angle $\theta$ used in $R_2$ is the same as that used in $R_1$, in general, no elements are annihilated by this similarity.

The elements in positions $(q,q)$, $(q,n+p)$, $(n+p,q)$, and $(n+p,n+p)$ (marked $*$) are affected by both $R_2^T$ from the left and by $R_2$ from the right, and none of them were made zero. This lead us to the equation
\[(a_{qq}^{(2)})^2 + (a_{n+p,n+p}^{(2)})^2 + 2(a_{q,n+p}^{(2)}) = (a_{qq}^{(1)})^2 + (a_{n+p,n+p})^2 + 2(a_{q,n+p}),
\]
which shows that there is liable to be some leakage of the weight from the diagonal entries $a_{qq}$ and $a_{n+p,n+p}$ into the off-diagonal entries $\hat{a}_{q,n+p}$ and $\hat{a}_{n+p,q}$ after the similarity. As we discussed before in Section 2.3.1, the overall off-diagonal norm of $A^{(1)}$ could possibly increase. This increase could, in theory, be large enough to offset the decrease produced by the similarity by the active rotation $R_1$.

Once again, the proof of convergence of an algorithm that use a transformation as in (2.37) as one of tools cannot depend on the strategy of showing that the off
diagonal norm decreases.

The computation of $Q$ is given in the following.

**function**: $Q = \text{Concentric}(A, p, n + q, \text{sort}, \text{tol})$

Given a real symmetric matrix $A \in \mathbb{R}^{2n \times 2n}$, integers $p$ and $n + q$ that satisfy $1 \leq p < q \leq n$, a Boolean parameter "sort", and $\text{tol} > 0$, this algorithm computes a real symplectic orthogonal transformation $Q \in \mathbb{R}^{2n \times 2n}$ as in (2.37) such that if $\hat{A} = Q^T A Q$, then $\hat{a}_{p,n+q} = \hat{a}_{n+q,p} = 0$. If sort = 0, then the order of the diagonal elements $(p,p)$ and $(n+q,n+q)$ is preserved, that is, $\text{sign}(a_{pp} - a_{n+q,n+q}) = \text{sign}(\hat{a}_{pp} - \hat{a}_{n+q,n+q})$. If sort is nonzero, then the sorting angle is used for ordering the diagonal elements $(p,p)$ and $(n+q,n+q)$ of $\hat{A}$ in decreasing order.

If $|a_{p,n+q}| > \text{tol}$ then

% compute small angle

$$\tau = \frac{a_{n+q,n+q} - a_{pp}}{2a_{p,n+q}}$$

$$t = \frac{\text{sign} \tau}{|\tau| + \sqrt{\tau^2 + 1}}$$
% as in (2.5)

if (sort $\neq$ 0 and $a_{pp} < a_{n+q,n+q}$) then

$$t = -\frac{1}{t}$$
% use large angle

endif

$$c = \frac{1}{\sqrt{t^2 + 1}} ; \quad s = tc$$

else

$$c = 1 ; \quad s = 0$$
endif
Q = I_{2n}; Q(p,p) = c; Q(p,n + q) = s; Q(n + q,p) = -s; Q(n + q,n + q) = c;
Q(qq) = c; Q(q,n + p) = s; Q(n + p,q) = -s; Q(n + p,n + p) = c

Note that an algorithm for setting \( \hat{a}_{n,n+p} = \hat{a}_{n+p,q} = 0 \), can be obtained by replacing \( p \) and \( n + q \) with \( q \) and \( n + p \), respectively in function Concentric.

2.4 Real 4 \times 4 Symplectic Orthogonals

Real 4 \times 4 symplectic orthogonals can all be expressed as products

\[
\begin{bmatrix}
  p_0 & -p_1 & -p_2 & -p_3 \\
  p_1 & p_0 & -p_3 & p_2 \\
  p_2 & p_3 & p_0 & -p_1 \\
  p_3 & -p_2 & p_1 & p_0
\end{bmatrix}
\begin{bmatrix}
  q_0 & 0 & q_2 & 0 \\
  0 & q_0 & 0 & q_2 \\
  -q_2 & 0 & q_0 & 0 \\
  0 & -q_2 & 0 & q_0
\end{bmatrix},
\] 

(2.43)

where \( p_0^2 + p_1^2 + p_2^2 + p_3^2 = 1 \) and \( q_0^2 + q_2^2 = 1 \) (see Mackey, Mackey, and Tisseur[26]).

Given \( 0 \neq x \in \mathbb{R}^4 \), consider the matrix

\[
G = \frac{1}{\sqrt{x^T x}}
\begin{bmatrix}
  x_1 & x_2 & x_3 & x_4 \\
  -x_2 & x_1 & x_4 & -x_3 \\
  -x_3 & -x_4 & x_1 & x_2 \\
  -x_4 & x_3 & -x_2 & x_1
\end{bmatrix}.
\] 

(2.44)

Then \( G \) is real symplectic orthogonal since it is of the form (2.43) with \( [p_0, p_1, p_2, p_3] = [x_1, -x_2, -x_3, -x_4]/\sqrt{x^T x} \) and \( [q_0, q_2] = [1, 0] \). The transformation \( G \) acts as a four-dimensional Givens rotation [8], [23], [26], that is, if \( y = Gx \), then \( y_2 = y_3 = y_4 = 0 \) and \( y_1 = \sqrt{x^T x} \); the norm of \( x \) moves into the first component of \( y \) by the transformation. Embedding \( G \) as a principal submatrix in rows and columns \( p,q,n + p,n + q \) of \( I_{2n} \) yields a real \( 2n \times 2n \) symplectic orthogonal matrix of the form.
where \([y_1, y_2, y_3, y_4] = [x_1, x_2, x_3, x_4]/\sqrt{x^Tx}\). This matrix can be used to zero out three out of four affected components of \(x \in \mathbb{R}^{2n}\).

Given a symmetric matrix \(A \in \mathbb{R}^{2n \times 2n}\), our aim is to find a real symplectic orthogonal matrix \(Q\) of the form as in (2.45) so that three new zeroes are obtained by the pre-multiplication \(QA\), and these introduced zeroes are preserved when the similarity is completed by post-multiplying by \(Q^T\). The pivot elements, marked by \(\diamond\) in Figure 2.5, should therefore be in positions \((q, r), (n + p, r), (n + q, r)\) for some \(r \notin \{p, q, n + p, n + q\}\), while the \((p, r)\) entry denoted by \(\ast\) is the one that receives all weight of those pivots after the transformation \(QA\).

To achieve this goal we set \(x = [A(p, r), A(q, r), A(n + p, r), A(n + q, r)]^T\). This vector \(x\) is called the target vector. Then construct the symplectic 4 \(\times\) 4 Givens rotation \(G\) for this vector \(x\) as in (2.44) and embed it as a principal submatrix of the \(2n \times 2n\) identity in rows (and columns) \(p, q, n + p, n + q\), the rows from which the entries of \(x\) were extracted, thus forming a real \(2n \times 2n\) symplectic orthogonal matrix \(Q\) as in (2.45).

The similarity \(QAQ^T\) can be viewed as the right multiplication on \(QA\) by \(Q^T\). This action affects only columns \(p, q, n + p, n + q\) of \(QA\). Since the zeroes introduced by \(QA\) are in column \(r \notin \{p, q, n + p, n + q\}\), these zeroes are preserved by right multiplication action by \(Q^T\). Three additional zeroes in the positions \((r, q), (r, n + p), (r, n + q)\) of \(QAQ^T\) are forced by symmetry of \(A\).

The computation of \(Q\) is given in the following algorithm. We will use the following notation adapted from [8]. If \(A\) is a \(2n \times 2n\) matrix, and \(v\) is a vector whose co-ordinates \(v_1, v_2, \ldots, v_k\) are positive integers between 1 to \(2n\), then \(A(v, i)\) denotes the vector \([A(v_1, i), \ldots, A(v_k, i)]\). Similarly, \(A(v, v)\) denotes the principal submatrix whose diagonal elements are \(A(v_1, v_1), A(v_2, v_2), \ldots, A(v_k, v_k)\). Givens(4, \(x\)) denotes the real 4 \(\times\) 4 symplectic Givens rotation \(G\) given in (2.44).
**Algorithm 1.** Given a real symmetric matrix \( A \in \mathbb{R}^{2n \times 2n} \) and integers \( p, q \) and \( r \) that satisfy \( 1 \leq p < q \leq n, \ r \notin \{p, q, n + p, n + q\} \), this algorithm computes a real symplectic orthogonal matrix \( Q \in \mathbb{R}^{2n \times 2n} \) as in (2.45) such that if \( \hat{A} = QAQ^T \), then \( \hat{a}_{qr} = \hat{a}_{n+p,r} = \hat{a}_{r,n+p} = \hat{a}_{r,n+q} = 0 \).

\[
v = [p, q, n + p, n + q]; \quad x = A(v, r) \\
Q = I_{2n}; \quad Q(v, v) = \text{Givens}(4, x)
\]

To be consistent with previous algorithms for applying the similarity, in this case we will use the similarity \( Q^T AQ \) in our algorithm.

### 2.5 Real Symplectic Householders

Let \( v \in \mathbb{R}^n \) be nonzero vector. An \( n \times n \) Householder reflector \( H \) is a symmetric orthogonal matrix of the form

\[
H = I - 2\frac{vv^T}{v^Tv}.
\]  

(2.46)

This matrix \( H \) was first introduced by Alston Scott Householder in 1958. The vector \( v \) is called Householder vector. The Householder reflector \( H \) can be used to zero selected components of a vector. In particular, suppose we are given a nonzero vector \( x \in \mathbb{R}^n \), a matrix \( H \) can be constructed to annihilate all but the first component of \( x \), that is, if \( y = Hx \), then \( y_2 = y_3 = \ldots = y_n = 0 \), and \( y_1 = \pm \|x\|_2 \), thus \( H \) moves norm of \( x \) onto the first component of \( x \). In this case \( x \)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
is called a target vector and the Householder vector \( v \) is defined by \( v = x \pm \|x\| e_1 \) where \( e_1 = [1 \ 0 \ 0 \ \ldots \ 0]^T \in \mathbb{R}^n \) [11]. The algorithm for computing Householder vector \( v \) can be found, for example, in [11, page 210]. Note that \( H \) is not a real symplectic matrix; it is not one of tools for our algorithms.

A real \( 2n \times 2n \) symplectic Householder matrix \( Q \) is a matrix of the form

\[
Q = \begin{bmatrix}
I_k & H \\
H & I_k
\end{bmatrix}_{2n \times 2n}
\]

(2.47)

where \( H \) is the real \((n - k) \times (n - k)\) Householder matrix as in (2.46). This matrix was described by Paige and Van Loan in [33]. \( Q \) is a real symplectic symmetric orthogonal matrix and it can be constructed to map \( n - k \) coordinates from either the first \( n \) or the last \( n \) coordinates of \( x \in \mathbb{R}^{2n} \) to a specific vector in \( \mathbb{R}^{n-k} \) [26].

In our algorithms, \( Q \) is designed to zero specified entries by performing a Givens-like action on \( A \) by \( Q^T = Q \) from the left in such a way that these introduced zeroes are not disturbed when the similarity is completed by post-multiplication by \( Q \). Since \( Q \) is of the form as in (2.47), pre-multiplication by \( Q^T(= Q) \) affects only rows \( k + 1 \) to \( n \) and rows \( n + k + 1 \) to \( 2n \), while post-multiplication by \( Q \) affects only columns \( k + 1 \) to \( n \) and columns \( n + k + 1 \) to \( 2n \). Thus, the coordinates of the target vector should be in the affected rows but not in the affected columns. The elements either in the positions \((k + 1, i),\ldots,(n, i)\) where \( 1 \leq i \leq k \) or \((n + k + 1, i),\ldots,(2n, i)\) where \( n + 1 \leq i \leq n + k \) can be the coordinates of the target vector. Figure 2.6 shows the action of pre-multiplication \( Q^T A \). Here the entries denoted by \( x \) form a target vector \( x \in \mathbb{R}^{n-k} \).

Figure 2.6: Left action by real symplectic Householder matrix

\[
\begin{array}{cccc}
\hline
k+1 & n & n+k+1 & 2n \\
\hline
k+1 & x & x & \\
n & x & & \\
n+k+1 & & & \\
2n & & & \\
\hline
\end{array}
\]
In this case, the Householder reflector $H \in \mathbb{R}^{(n-k) \times (n-k)}$ is designed so that

$$Hx = H \begin{bmatrix} x \\ x \\ \vdots \\ x \end{bmatrix} = \begin{bmatrix} \bullet \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \|x\| \mathbf{e}_1 \in \mathbb{R}^{n-k}.$$  

This left action moves norm of $x$ onto the $(k+1,i)$ entry (marked $\bullet$). Note that rows 1 to $k$ and $n+1$ to $n+k$ do not change since these rows are acted on by $I_k$.  

To complete the similarity, we must perform the post-multiplication of $Q^T A$ by $Q$. This action will effect only columns $k+1$ to $n$ and $n+k+1$ to $2n$. Since $I_k$ acts on columns 1 to $k$ and columns $n+1$ to $n+k$, these columns are not change. Since all introduced zeroes are in column $i$ where $1 \leq i \leq k$, these zeroes are not destroyed by this action, and additional zeroes are forced to appear in the positions $(i,k+2), \ldots, (i,n)$ by symmetry.
Chapter 3

Real Symplectic Bird Form

In this Chapter, we present a structured condensed form for real symplectic symmetric matrices that is achievable in a finite number of steps. This condensed form, which we called the *real symplectic bird form*, is defined as a real symplectic symmetric matrix of the form

\[
\begin{bmatrix}
E & F \\
F^T & H
\end{bmatrix}
\]  

(3.1)

where \(E, F, H \in \mathbb{R}^{n \times n}\), \(E\) is symmetric tridiagonal, \(F\) is lower triangular, and \(H\) is symmetric. This condensed form was described earlier by Bunse-Gerstner, Byers, and Mehrmann in [3].

Let \(A\) be a real skew-Hamiltonian matrix. Then \(A\) can be written as \([D \ V] \ W^{-1}\) where \(U, V \in \mathbb{R}^{n \times n}\) are skew-symmetric. In [37], Van Loan described how a real symplectic orthogonal \(Q\) can be constructed such that \(Q^T AQ\) is in structured Hessenberg form, that is,

\[
Q^T AQ = \begin{bmatrix} H & R \\ 0 & H^T \end{bmatrix},
\]

(3.2)

where \(H \in \mathbb{R}^{n \times n}\) is upper Hessenberg and \(R \in \mathbb{R}^{n \times n}\) is symmetric. The transformations used in Van Loan’s procedure are real \(2n \times 2n\) symplectic Householders of the form as in (2.47) and real \(2n \times 2n\) symplectic plane rotations as in (2.17). This reduction can be accomplished in a finite number of steps. In [8], Faßbender, Mackey, and Mackey gave another method to achieve the structured Hessenberg form for real skew-Hamiltonian matrices. In this method, a finite sequence of \(4 \times 4\) based symplectic Givens as in (2.45) and \(2 \times 2\) based symplectic Givens as in (2.17) is used for the reduction.

Each of these methods can be adapted in a straightforward way to reduce a real symplectic symmetric matrix to the real symplectic bird form given in (3.1). These adaptations are the subject of the next sections.
3.1 Via Symplectic Householders

Now we show how Van Loan’s method [37] can be used to reduce a real symplectic symmetric matrix to the real symplectic bird form in a finite number of steps. Before presenting this method in general, we illustrate this method on a typical real $8 \times 8$ symplectic symmetric matrix $A$. This is done in Figure 3.1.

Here the labels $House$ and $2 \times 2$ $Givens$ indicate similarity by an appropriate real $8 \times 8$ symplectic Householder matrix of the form as in (2.47) and real $8 \times 8$ symplectic plane rotation as in (2.17), respectively.

\[
\begin{bmatrix}
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\end{bmatrix}
\xrightarrow{House}
\begin{bmatrix}
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\end{bmatrix}
\xrightarrow{2 \times 2 \text{Givens}}
\begin{bmatrix}
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\end{bmatrix}
\xrightarrow{House}
\begin{bmatrix}
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & 0 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\end{bmatrix}
\xrightarrow{House}
\begin{bmatrix}
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\end{bmatrix}
\xrightarrow{House}
\begin{bmatrix}
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\end{bmatrix}
\]
Let us consider the first two steps from Figure 3.1 in some detail. In the first step, the entries denoted by $x$ in the first column of $A$ form a target vector $x \in \mathbb{R}^3$. This vector $x$ is used to compute the Householder vector $v \in \mathbb{R}^3$ which determines the $3 \times 3$ Householder matrix $H$ and thereby the real $8 \times 8$ symplectic Householder matrix as in (2.47). Pre-multiplication $Q^T A$ affects rows 2, 3, 4, and rows 6, 7, 8 of $A$ introducing zeroes in positions (7, 1) and (8, 1), which are the second and third components of the target vector $x$. The norm of $x$ moves to the first component of $x$, the (5, 1) entry, denoted by $\bullet$. Right multiplication of $Q^{(1)T} A$ by $Q^{(1)}$ affects only columns 2, 3, 4, and columns 6, 7, 8 of $Q^{(1)T} A$; the zeroes introduced by the left action of $Q^{(1)T}$ are not destroyed and two additional zeroes appear, by symmetry.

In the second step of the reduction, the two elements marked by $x$ form a target vector; they are coupled during the transformation. One of these (the (6, 1) entry) is the pivot element and it is made zero by a real symplectic plane rotation.
$Q^{(2)T}$ acting from the left. The entire weight of the pivot moves to its partner, the $(2, 1)$ entry, by this action. Since the left multiplication of $A^{(1)}$ by $Q^{(2)}$ affects rows 2 and 6, it will not destroy the zeroes introduced in the first step of the reduction. To complete the similarity, we must postmultiply $Q^{(2)T}A^{(1)}$ by $Q^{(2)}$. This action affects columns 2 and 6; the zeroes introduced by the left action of $Q^{(2)T}$ and by the first step of the reduction are preserved and one additional zero in the position $(1, 6)$ is forced by symmetry.

Subsequent steps of our reduction are depicted analogously in Figure 3.1. The entries denoted by $\times$ form a target vector. A real Householder matrix or real $2 \times 2$ plane rotation is constructed from this vector and then appropriately embedded in $I_8$ to give a real $8 \times 8$ symplectic Householder or a real $8 \times 8$ symplectic plane rotation $Q^{(1)}$. Each similarity by $Q^{(1)}$ adds new zero entries to the diagram. The entry denoted by $\bullet$ carries all the norm of the target vector.

As can be seen from Figure 3.1, two House and one $2 \times 2$ Givens are needed to complete the reduction in columns 1 and 2, while only one $2 \times 2$ Givens is needed for column 3. Thus, the total number of transformations needed for the reduction of this typical $8 \times 8$ matrix $A$ to real symplectic bird form is seven transformations.

In general, the reduction of a real $2n \times 2n$ symplectic symmetric matrix to the real symplectic bird form using real symplectic Householders and real $2n \times 2n$ symplectic plane rotations is given in the following algorithm. To complete the reduction, three transformations are needed (two House and one $2 \times 2$ Givens) for each column $i$ for $i = 1, 2, \ldots, n - 2$, and only one $2 \times 2$ Givens needed for column $n - 1$. The total number of transformations needed for completing the reduction is $3n - 5$.

We first include the functions house, row.house, and col.house taken from [10, page 196] which will be called by our algorithm. Here length($x$) specifies dimension of vector $x$.

**function: $v =$ house($x$)**

Given $x \in \mathbb{R}^n$, this function computes $v \in \mathbb{R}^n$ with $v(1) = 1$ such that $(I - 2\frac{vv^T}{v^Tv})x$ is zero in all but the first component.

$n = \text{length}(x); \mu = \|x\|_2; v = x$

if $\mu \neq 0$

$\beta = x(1) + \text{sign}(x(1))\mu; v(2 : n) = \frac{v(2 : n)}{\beta}$

endif

$v(1) = 1$

**function: $A =$ row.house($A, v$)**

Given $A \in \mathbb{R}^{m \times n}$ and $v \in \mathbb{R}^m$ with $v(1) = 1$, this algorithm overwrites $A$ with $HA$
function: $A = \text{col.house}(A, v)$

Given $A \in \mathbb{R}^{m \times n}$ and $v \in \mathbb{R}^n$ with $v(1) = 1$, this algorithm overwrites $A$ with $AH$

where $H = I - 2\frac{vv^T}{v^Tv}$.

$$\beta = -\frac{2}{v^Tv}$$
$$w = \beta A^Tv$$
$$A = A + vw^T$$

The reduction of a real $2n \times 2n$ symplectic symmetric matrix to real symplectic bird form can be accomplished by the following algorithm.

**Algorithm 2.** Given a real symplectic symmetric matrix $A \in \mathbb{R}^{2n \times 2n}$, this algorithm overwrites $A$ with $\hat{A} = Q^TAQ$ where $Q \in \mathbb{R}^{2n \times 2n}$ is real symplectic orthogonal and $\hat{A}$ is in symplectic bird form as in (3.1). Here zeroes are created by either real symplectic Householders as in (2.47) or real symplectic plane rotations as in (2.17) acting from the left.

$$Q = I_{2n}$$

for $q = 1 : n - 2$

$\quad x = A(n + q + 1 : 2n, q)$  \hspace{1cm} \% target vector
$\quad v = \text{house}(x)$  \hspace{1cm} \% Householder vector

$\quad %$ update rows $n + q + 1, \ldots, 2n$ of $A$
$\quad %$ introduce zeroes in positions $(n + q + 2, q), \ldots, (2n, q)$
$\quad A(n + q + 1 : 2n, q : 2n) = \text{row.house}(A(n + q + 1 : 2n, q : 2n), v)$

$\quad %$ update rows $q + 1, \ldots, n$ of $A$
$\quad A(q + 1 : n, q : 2n) = \text{row.house}(A(q + 1 : n, q : 2n), v)$

$\quad %$ update columns $n + q + 1, \ldots, 2n$ of $A$
$\quad %$ introduce zeroes in positions $(q, n + q + 2), \ldots, (q, 2n)$
$\quad A(q : 2n, n + q + 1 : 2n) = \text{col.house}(A(q : 2n, n + q + 1 : 2n), v)$

$\quad %$ update columns $q + 1, \ldots, n$ of $A$
$\quad A(q : 2n, q + 1 : n) = \text{col.house}(A(q : 2n, q + 1 : n), v)$

$\quad %$ update columns $q + 1, \ldots, n$ and $n + q + 1, \ldots, 2n$ of $Q$

36

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
\[ Q(q : 2n, q + 1 : n) = \text{col.house}(Q(q : 2n, q + 1 : n), v) \]
\[ Q(q : 2n, n + q + 1 : 2n) = \text{col.house}(Q(q : 2n, n + q + 1 : 2n), v) \]
\[ \tilde{Q} = \text{Givens.left}(A, n + q + 1, q) \]
\[ A = \tilde{Q}^T A \tilde{Q} ; \quad Q = \tilde{Q} \tilde{Q} \]
\[ x = A(q + 1 : n, q) \quad \text{% target vector} \]
\[ v = \text{house}(x) \quad \text{% Householder vector} \]

\% update rows \( q + 1, \ldots, n \) of \( A \)
\% introduce zeroes in positions \((q + 2, q), \ldots, (n, q)\)
\[ A(q + 1 : n, q : 2n) = \text{row.house}(A(q + 1 : n, q : 2n), v) \]

\% update rows \( n + q + 1, \ldots, 2n \) of \( A \)
\[ A(n + q + 1 : 2n, q : 2n) = \text{row.house}(A(n + q + 1 : 2n, q : 2n), v) \]

\% update columns \( q + 1, \ldots, n \) of \( A \)
\% introduce zeroes in positions \((q, q + 2), \ldots, (q, n)\)
\[ A(q : 2n, q + 1 : n) = \text{col.house}(A(q : 2n, q + 1 : n), v) \]

\% update columns \( n + q + 1, \ldots, 2n \) of \( A \)
\[ A(q : 2n, n + q + 1 : 2n) = \text{col.house}(A(q : 2n, n + q + 1 : 2n), v) \]

\% update columns \( q + 1, \ldots, n \) and \( n + q + 1, \ldots, 2n \) of \( Q \)
\[ Q(q : 2n, q + 1 : n) = \text{col.house}(Q(q : 2n, q + 1 : n), v) \]
\[ Q(q : 2n, n + q + 1 : 2n) = \text{col.house}(Q(q : 2n, n + q + 1 : 2n), v) \]

\textit{endfor}
\[ \tilde{Q} = \text{Givens.left}(A, 2n, n - 1) \]
\[ A = \tilde{Q}^T A \tilde{Q} ; \quad Q = \tilde{Q} \tilde{Q} \]

### 3.2 Via 4 x 4 Symplectic Givens

We now show how 4 x 4 based symplectic Givens matrices as in (2.45) and 2 x 2 based symplectic Givens matrices as in (2.17) can be used to reduce a real 2n x 2n symplectic symmetric matrix to the real symplectic bird form in a finite number of steps. An illustration when 2n = 8 is given in Figure 3.2. Here the label 4 x 4 \textit{Givens} and 2 x 2 \textit{Givens} indicate similarity by an appropriate matrix of the form as in (2.45) and (2.17), respectively.

Let us now examine the first step from Figure 3.2 in some detail. The four entries denoted by \( \times \) in the first column of \( A \) form a target vector \( x \in \mathbb{R}^4 \) for the first step of reduction. This vector \( x \) is used to construct a real 4 x 4 symplectic Givens rotation \( G \) as in (2.44), and then form a real 8 x 8 symplectic orthogonal matrix \( Q \) by embedding \( G \) as a principal submatrix of the 8 x 8 identity in rows (and columns) 2, 3, 6 and 7, the rows from which the entries of \( x \) were extracted.
Pre-multiplication $QA$ affects rows 2, 3, 6 and 7, and introduces three zeroes into the $x$ entries, moving all the norm of $x$ to the entry labelled $\bullet$. To preserve the symmetry of $A$, the similarity $QAQ^T$ is performed. Right multiplication of $QA$ by $Q^T$ affects only columns 2, 3, 6 and 7; it will not destroy the introduced zeroes by $Q$ since these zeroes are in the first column of $QA$, and this action also makes three additional zeroes by symmetry. Subsequent steps of our reduction are represented analogously in Figure 3.2.

From Figure 3.2, it can be seen that two $4 \times 4$ Givens are needed to complete...
the reduction in the first column. The target vectors for these transformations are formed from the elements in the positions $(2, 1), (3, 1), (6, 1), (7, 1)$ and $(2, 1), (4, 1), (6, 1), (8, 1)$, respectively. In the second column, only one $4 \times 4$ Givens is needed; the target vector for this transformation is formed by the elements in positions $(3, 2), (4, 2), (7, 2), (8, 2)$. In the last column, only one $2 \times 2$ Givens is needed with the target vector formed by the $(4, 3)$ and $(8, 3)$ entries. Thus, the total number of transformations needed for the reduction of a $8 \times 8$ matrix $A$ is 4 transformations.

In general, the reduction of a real $2n \times 2n$ symplectic symmetric matrix to real symplectic bird form is given in the following algorithm. To complete the reduction in the first column, we must perform $n - 2$ similarities using $4 \times 4$ Givens. The target vectors needed for these similarities are formed from the elements in the positions in the following order:

\[(2, 1), (3, 1), (n + 2, 1), (n + 3, 1);\]
\[(2, 1), (4, 1), (n + 2, 1), (n + 4, 1);\]
\[\vdots\]
\[(2, 1), (n, 1), (n + 2, 1), (2n, 1).\]

The reduction of the second column is completed with $n - 3$ similarities using $4 \times 4$ Givens. The target vectors for these transformations are formed from the elements in the positions in the following order:

\[(3, 2), (4, 2), (n + 3, 2), (n + 4, 2);\]
\[(3, 2), (5, 2), (n + 3, 2), (n + 5, 2);\]
\[\vdots\]
\[(3, 2), (n, 2), (n + 3, 2), (2n, 2).\]

Thus, we can conclude that the number of similarities using $4 \times 4$ Givens for the reduction of each column $i$ for $i = 1, 2, \ldots, n - 2$ is $n - i - 1$ similarities. Only one $2 \times 2$ Givens is needed for the reduction of the column $n - 1$. The total number of the similarities for the reduction of a real $2n \times 2n$ symplectic symmetric matrix to the real symplectic bird form is

\[\sum_{i=1}^{n-2} (n - i - 1) = \frac{1}{2}(n^2 - 3n + 2).\]

To be consistent with previous algorithms for applying the similarity, in this case we will use the similarity $Q^T AQ$ in our algorithm.

**Algorithm 3.** Given a real symplectic symmetric matrix $A \in \mathbb{R}^{2n \times 2n}$, this algorithm overwrites $A$ with $\hat{A} = Q^T AQ$ where $Q \in \mathbb{R}^{2n \times 2n}$ is real symplectic orthogonal and $\hat{A}$ is in symplectic bird form as in (3.1). Here zeroes are created by either $4 \times 4$ based symplectic Givens matrices as in (2.45) or real symplectic plane rotations as in (2.17), both acting from the left. Algorithm 1 is used to compute a real $4 \times 4$ based symplectic Givens matrix as in (2.45).
\[ Q = I_{2n} \]
for \( p = 2 : n - 1 \)
  for \( q = p + 1 : n \)
    Use Algorithm 1 to find \( Q \) such that \( \tilde{A}_{q,p-1} = \tilde{A}_{n+p,p-1} = \tilde{A}_{n+q,p-1} = 0 \)
    \[ \tilde{A} = \tilde{Q}^T A \tilde{Q} ; \quad A = \tilde{A} ; \quad Q = Q \tilde{Q} \]
  endfor
endfor

\[ \tilde{Q} = \text{Givens.left}(A, 2n, n-1) \] % 2 x 2 Givens
\[ \tilde{A} = \tilde{Q}^T A \tilde{Q} ; \quad A = \tilde{A} ; \quad Q = Q \tilde{Q} \]
Chapter 4
Computing the Real Symplectic SVD: Using the Polar Decomposition

Let $A$ be a real $2n \times 2n$ symplectic matrix with polar decomposition $A = UH$ where $U$ is real symplectic orthogonal and $H$ is real symmetric positive definite. As was discussed in Chapter 1, the problem of finding structure preserving algorithms for computing a real symplectic SVD of $A$ can be reduced to finding structure preserving algorithms for computing a structured spectral decomposition of $H$. That is, computing a real symplectic SVD of $A$ comprises two steps: a step in which the real symplectic orthogonal $U$ is computed, and a step in which the structured spectral decomposition of the real symplectic symmetric $H$ is computed. Recently, structure preserving iterations for computing the symplectic unitary polar factor of $A$ were given in [16]. In our work, the real symplectic orthogonal $U$ is computed by using the well-known Newton iteration:

$$X_{k+1} = \frac{1}{2}(X_k + X_k^{-T}), \quad X_0 = A.$$  \hspace{1cm} (4.1)

As was discussed in [16], the Newton iteration is not structure-preserving for the real symplectic group $Sp(2n, \mathbb{R})$ since this group is not closed under addition. However, numerical experiments in [16] showed that though the Newton iterations destroy symplectic structure on the first iteration, they gradually restore it since the limit $U$ is symplectic. In our experiments, the number of iterations needed by the Newton iteration to converge to $U$ is computed and the departure of $U$ from being real symplectic and orthogonal is also measured.

After the polar factor $U$ was computed, the real symplectic symmetric factor $H$ can be determined by $H = U^T A$. To complete the process of finding a real symplectic SVD, structure preserving algorithms for computing a structured spectral decomposition of $H$ must be developed. The main goal of this section is to
find such algorithms.

For the $4 \times 4$ case, we present a finite step algorithm for computing a structured spectral decomposition in Section 4.1. The subsequent sections deal with the $2n \times 2n$ case for $n > 2$. Note that all the tools used by our algorithms were described in Chapter 2.

### 4.1 $4 \times 4$ Case

Given a real $4 \times 4$ symplectic symmetric matrix

$$A = \begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
a_{12} & a_{22} & a_{23} & a_{24} \\
a_{13} & a_{23} & a_{33} & a_{34} \\
a_{14} & a_{24} & a_{34} & a_{44}
\end{bmatrix},$$

then

$$A^* = -JA^T J$$

where

$$J = \begin{bmatrix}
0 & I_2 \\
-I_2 & 0
\end{bmatrix},$$

and

$$A^* = \begin{bmatrix}
a_{33} & a_{34} & -a_{13} & -a_{23} \\
a_{34} & a_{44} & -a_{14} & -a_{24} \\
-a_{13} & -a_{14} & a_{11} & a_{12} \\
-a_{23} & -a_{24} & a_{12} & a_{22}
\end{bmatrix}.$$

The following useful relations which follow from the symplectic property of $A$, i.e. $AA^* = I_4$, will be used later. Note that the pair to the left of each equation denotes the entry of $AA^* = I_4$.

\begin{align*}
(1, 2) : & \quad a_{11}a_{34} + a_{12}a_{44} - a_{13}a_{14} - a_{14}a_{24} = 0, \quad (4.2) \\
(2, 1) : & \quad a_{12}a_{33} + a_{22}a_{34} - a_{23}a_{13} - a_{24}a_{23} = 0, \quad (4.3) \\
(2, 3) : & \quad -a_{13}a_{12} - a_{14}a_{22} + a_{11}a_{23} + a_{12}a_{24} = 0, \quad (4.4) \\
(3, 2) : & \quad a_{13}a_{34} + a_{23}a_{44} - a_{33}a_{14} - a_{34}a_{24} = 0. \quad (4.5)
\end{align*}

To show that a structured spectral decomposition of $A$ can be always accomplished in finite steps, we begin by giving the following useful propositions.

**Proposition 4.1.** Let $A \in \mathbb{R}^{4 \times 4}$ be a real symplectic symmetric matrix of the form

$$A = \begin{bmatrix}
a_{11} & a_{12} & 0 & a_{14} \\
a_{12} & a_{22} & a_{23} & 0 \\
0 & a_{23} & a_{33} & a_{34} \\
a_{14} & 0 & a_{34} & a_{44}
\end{bmatrix}, \quad \text{with} \quad a_{12} \neq 0,$$
and let $Q$ be a real $4 \times 4$ symplectic double rotation of the form

$$Q = \begin{bmatrix}
  c & s & 0 & 0 \\
  -s & c & 0 & 0 \\
  0 & 0 & c & s \\
  0 & 0 & -s & c
\end{bmatrix}, \quad \text{where } c = \cos \theta \quad s = \sin \theta.$$

Let $\hat{A} = [\hat{a}_{ij}] = Q^T AQ$. Then

1. $\hat{a}_{13} = -\hat{a}_{24}$

2. If $\theta$ is chosen to zero out the $(1,2)$ and $(2,1)$ entries of $\hat{A}$, then the $(3,4)$ and $(4,3)$ entries of $\hat{A}$ are also zeroed out, that is,

$$\hat{A} = Q^T AQ = \begin{bmatrix}
  \cdot & \cdot & \cdot & \cdot \\
  \cdot & \cdot & \cdot & 0 \\
  \cdot & \cdot & 0 & \cdot \\
  \cdot & 0 & \cdot & \cdot
\end{bmatrix}.$$

Proof. Writing $A$ and $Q$ in block form:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{12}^T & A_{22} \end{bmatrix},$$

where $A_{11} = \begin{bmatrix} a_{11} & a_{12} \\ a_{12} & a_{22} \end{bmatrix}$, $A_{12} = \begin{bmatrix} 0 & a_{14} \\ a_{23} & 0 \end{bmatrix}$, $A_{22} = \begin{bmatrix} a_{33} & a_{34} \\ a_{34} & a_{44} \end{bmatrix}$, and

$$Q = \begin{bmatrix} R & 0 \\ 0 & R \end{bmatrix}, \quad \text{where } R = \begin{bmatrix} c & s \\ -s & c \end{bmatrix},$$

we have that

$$\hat{A} = [\hat{a}_{ij}] = Q^T AQ = \begin{bmatrix} R^T A_{11} R & R^T A_{12} R \\ R^T A_{12}^T R & R^T A_{22} R \end{bmatrix}. \quad (4.6)$$

As we discussed in Section 2.3.1, each block undergoes the same similarity transformation. Since trace($A_{12}$) = 0 and any similarity is trace preserving, we get $\hat{a}_{13} = -\hat{a}_{24}$ from the block $R^T A_{12} R$.

Since $AA^* = I_4$, the elements in the $(1,2)$ and $(2,1)$ positions of $AA^*$ are zeroes. Putting $a_{13} = a_{24} = 0$ in (4.2) and (4.3) yields

$$a_{12}a_{33} + a_{22}a_{34} = a_{11}a_{34} + a_{12}a_{44} = 0. \quad (4.7)$$

Here we have used the hypothesis that $a_{13} = a_{31} = a_{24} = a_{42} = 0$. Since $a_{12} \neq 0$ by assumption, from (4.7) we have

$$a_{33} - a_{44} = \frac{a_{34}}{a_{12}}(a_{11} - a_{22}). \quad (4.8)$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Computing \( \hat{a}_{34} \) in the block submatrix \( RT A_{22} R \) and using the symmetry of \( A_{22} \), we obtain

\[
\hat{a}_{34} = \hat{a}_{43} = a_{34}(c^2 - s^2) + (a_{33} - a_{44})cs
\]  

(4.9)

Then, combining (4.8) and (4.9), we have

\[
\hat{a}_{34} = a_{34}(c^2 - s^2) + \frac{a_{34}}{a_{12}}(a_{11} - a_{22})cs,
\]

\[
= \frac{a_{34}}{a_{12}} \{ a_{12}(c^2 - s^2) + (a_{11} - a_{22})cs \},
\]

\[
= \frac{a_{34}}{a_{12}} \hat{a}_{12}.
\]

But \( \hat{a}_{12} \) was made zero, hence \( \hat{a}_{34} = 0 \). Therefore we can conclude that

\[
\hat{A} = Q^T AQ = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

, with \( \hat{a}_{13} = -\hat{a}_{24} \).

\[\square\]

A useful similarity that further reduces the matrix \( \hat{A} \) from Proposition 4.1 is given in the next proposition.

**Proposition 4.2.** Let \( A \in \mathbb{R}^{4 \times 4} \) be a real symplectic symmetric matrix of the form

\[
A = \begin{bmatrix}
a_{11} & a_{13} & a_{14} \\
a_{13} & a_{22} & a_{24} \\
a_{14} & a_{24} & a_{33} \\
a_{13} & a_{23} & a_{33} & a_{34} \\
\end{bmatrix}
\]

and let \( Q \) be a real \( 4 \times 4 \) symplectic double plane rotation of the form

\[
Q = \begin{bmatrix}
c & 0 & 0 & s \\
0 & c & s & 0 \\
0 & -s & c & 0 \\
-s & 0 & 0 & c \\
\end{bmatrix}, \text{ where } c = \cos \theta, s = \sin \theta.
\]

Let \( \hat{A} = [\hat{a}_{ij}] = Q^T AQ \). Then

1. The entries in positions (1,3), (3,1), (2,4), and (4,2) of \( A \) are preserved, and the original zeroes in positions (1,2), (2,1), (3,4), (4,3) of \( A \) are not destroyed by this similarity.

2. If \( \theta \) is chosen to zero out the (1,4) and (4,1) entries of \( \hat{A} = Q^T AQ \). Then the entries (2,3) and (3,2) of \( \hat{A} \) are also zeroed out, that is,

\[
\hat{A} = Q^T AQ = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\end{bmatrix}
\]
Proof. A straightforward computation of \( \hat{A} \) shows that

\[
\begin{align*}
\hat{a}_{13} &= \hat{a}_{31} = a_{13}(c^2 + s^2) = a_{13}, \\
\hat{a}_{24} &= \hat{a}_{42} = -a_{13}(c^2 + s^2) = -a_{13}, \\
\hat{a}_{12} &= \hat{a}_{21} = -(a_{13} + a_{24})cs, \\
\hat{a}_{34} &= \hat{a}_{43} = (a_{13} + a_{24})cs.
\end{align*}
\]

(4.10)

(4.11)

(4.12)

(4.13)

That the (1,3) entry of \( A \) is preserved follows immediately from (4.10). The hypothesis that \( a_{13} = -a_{24} \) together with (4.11) implies that the (2,4) entry of \( A \) is also preserved. Again, since \( a_{13} + a_{24} = 0 \) by hypothesis, from (4.12) and (4.13) we conclude that \( \hat{a}_{12} = \hat{a}_{21} = \hat{a}_{34} = \hat{a}_{43} = 0 \).

Since \( AA^* = I_4 \), the elements in the (2,3) and (3,2) positions of \( AA^* \) are zeroes. Putting \( a_{12} = a_{34} = 0 \) in (4.4) and (4.5) yields

\[
a_{11}a_{23} - a_{14}a_{22} = a_{23}a_{44} - a_{33}a_{14} = 0. \tag{4.14}
\]

Here we have used the hypothesis that \( a_{12} = a_{21} = a_{34} = a_{43} = 0 \). Since \( a_{14} \neq 0 \) by assumption, from (4.14) we obtain

\[
a_{22} - a_{33} = \frac{a_{23}}{a_{14}}(a_{11} - a_{44}). \tag{4.15}
\]

Once again, the computation of \( \hat{A} \) gives

\[
\hat{a}_{23} = a_{23}(c^2 - s^2) + (a_{22} - a_{33})cs. \tag{4.16}
\]

Combining (4.15) and (4.16), one obtains

\[
\frac{\hat{a}_{23}}{a_{14}} = \frac{a_{23}}{a_{14}}(c^2 - s^2) + \frac{a_{23}}{a_{14}}(a_{11} - a_{44})cs,
\]

\[
= \frac{a_{23}}{a_{14}} \left\{ a_{14}(c^2 - s^2) + (a_{11} - a_{44})cs \right\},
\]

\[
= \frac{a_{23}}{a_{14}} \hat{a}_{14}.
\]

Therefore, choosing \( Q \) so that \( \hat{a}_{14} = 0 \), also yields \( \hat{a}_{23} = 0 \). Hence, we can conclude that the matrix \( \hat{A} \) is of the form

\[
\hat{A} = Q^T AQ = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}, \quad \text{with} \quad \hat{a}_{13} = -\hat{a}_{24} = a_{13}. \quad \square
\]

We now show how a structured spectral decomposition of \( A = A^{(0)} \) can be computed by a sequence of at most 6 structure-preserving similarity transformations. Each of these similarity transformations is denoted by

\[
A^{(\ell+1)} = Q^{(\ell)T} A^{(\ell)} Q^{(\ell)}, \tag{4.17}
\]

45
where $Q^{(i)} \in SpO(4, \mathbb{R})$ and illustrated in the following figures. Note that all computation steps presented are exactly arithmetic. The entries represented by $\diamond$ represent the targets for annihilation ( pivots) at each step, while $\times$ is used to denote the other entries of matrix.

**Step 1.** If $a_{13}^{(0)} = 0$, then we go to Step 2. If not, we use a real symplectic plane rotation $Q^{(0)} = \begin{bmatrix} c & 0 & s & 0 \\ 0 & 1 & 0 & 0 \\ -s & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ to zero the $(1,3)$ and $(3,1)$ entries of $A^{(1)} = Q^{(0)^T} A^{(0)} Q^{(0)}$.

\[
\begin{bmatrix}
\times & \times & \diamond & \times \\
\times & \times & \times & \times \\
\diamond & \times & \times & \times \\
\times & \times & \times & \times \\
\end{bmatrix} \rightarrow \begin{bmatrix}
\times & \times & 0 & \times \\
\times & \times & \times & \times \\
0 & \times & \times & \times \\
\times & \times & \times & \times \\
\end{bmatrix}
\]

\[A^{(0)} \rightarrow A^{(1)}\]

**Step 2.** If $a_{24}^{(1)} = 0$, then we go to Step 3. Otherwise, we use a real symplectic plane rotation $Q^{(1)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -s \end{bmatrix}$ to zero the $(2,4)$ and $(4,2)$ entries of $A^{(2)} = Q^{(1)^T} A^{(1)} Q^{(1)}$. Note that this will not destroy the zero in the $(1,3)$ and $(3,1)$ positions created in the first step since the similarity only affects rows and columns 2,4 of $A^{(4)}$.

\[
\begin{bmatrix}
\times & \times & 0 & \times \\
\times & \times & \times & \times \\
0 & \times & \times & \times \\
\times & \times & \times & \times \\
\end{bmatrix} \rightarrow \begin{bmatrix}
\times & \times & 0 & \times \\
\times & \times & \times & \times \\
0 & \times & \times & \times \\
\times & \times & \times & \times \\
\end{bmatrix}
\]

\[A^{(1)} \rightarrow A^{(2)}\]

**Step 3.** In this step, we consider the following 2 cases.

**Case 1.** $a_{12}^{(2)} \neq 0$.

Use a real symplectic double rotation $Q^{(2)} = \begin{bmatrix} c & s & 0 & 0 \\ -s & c & 0 & 0 \\ 0 & 0 & c & s \\ 0 & 0 & 0 & -s \end{bmatrix}$ to zero the $(1,2)$ and $(2,1)$ entries of $A^{(3)} = Q^{(2)^T} A^{(2)} Q^{(2)}$. By Proposition 4.1, the entries $(3,4)$ and $(4,3)$ of $A^{(3)}$ also become zero, and $a_{13}^{(3)} = -a_{24}^{(3)}$.

\[
\begin{bmatrix}
\times & \diamond & 0 & \times \\
\diamond & \times & \times & 0 \\
0 & \times & \times & \times \\
\times & 0 & \times & \times \\
\end{bmatrix} \rightarrow \begin{bmatrix}
\times & 0 & \times & \times \\
0 & \times & \times & \times \\
\times & \times & \times & 0 \\
\times & \times & 0 & \times \\
\end{bmatrix}
\]

\[A^{(2)} \rightarrow A^{(3)}\]
Case 2. $a_{12}^{(2)} = 0$.

Using $a_{12}^{(2)} = a_{13}^{(2)} = a_{24}^{(2)} = 0$, equations (4.2) and (4.3) give $a_{11}^{(2)}a_{34}^{(2)} = a_{22}^{(2)}a_{34}^{(2)} = 0$. If $a_{34}^{(2)} \neq 0$, then $a_{11}^{(2)} = a_{22}^{(2)} = 0$, so $A^{(2)}$ is of the form

$$A^{(2)} = \begin{bmatrix}
0 & 0 & 0 & \times \\
0 & 0 & \times & 0 \\
\times & \times & \times & \\
\times & 0 & \times & \times
\end{bmatrix}.$$

Use a real symplectic double rotation $Q^{(2)} = \begin{bmatrix} c & 0 & 0 & -s \\
-s & c & 0 & 0 \\
0 & 0 & c & s \\
0 & 0 & -s & c
\end{bmatrix}$ to zero the $(3, 4)$ and $(4, 3)$ entries of $A^{(3)} = Q^{(2)T}A^{(2)}Q^{(2)}$. Then $a_{13}^{(3)} = -a_{24}^{(3)}$.

$$A^{(2)} \rightarrow A^{(3)}$$

Step 4. In this step, we have 3 cases as follows:

**Case 1.** $a_{14}^{(3)} \neq 0$ and either $A^{(3)} = \begin{bmatrix} x & 0 & x & x \\
x & 0 & x & x \\
x & x & 0 & x \\
x & x & 0 & x
\end{bmatrix}$ or $A^{(3)} = \begin{bmatrix} 0 & 0 & x & x \\
0 & 0 & x & x \\
x & x & 0 & x \\
x & x & 0 & x
\end{bmatrix}$.

Use a real symplectic double rotation $Q^{(3)} = \begin{bmatrix} c & 0 & 0 & s \\
0 & c & 0 & s \\
0 & 0 & c & s \\
0 & 0 & -s & c
\end{bmatrix}$ to zero the $(1, 4)$ and $(4, 1)$ entries of $A^{(4)} = Q^{(3)T}A^{(3)}Q^{(3)}$. By Proposition 4.2, the entries $(2, 3)$ and $(3, 2)$ of $A^{(4)}$ also become zero, and all zeroes in the positions $(1, 2), (2, 1), (3, 4), (4, 3)$ are preserved.

$$A^{(3)} \rightarrow A^{(4)}$$

**Case 2.** $a_{14}^{(3)} = 0$ and $A^{(3)} = \begin{bmatrix} x & 0 & x & x \\
x & 0 & x & x \\
x & x & 0 & x \\
x & x & 0 & x
\end{bmatrix}$.

Using $a_{12}^{(3)} = a_{13}^{(3)} = a_{34}^{(3)} = 0$, equations (4.4) and (4.5) give $a_{11}^{(3)}a_{23}^{(3)} = a_{23}^{(3)}a_{44}^{(3)} = 0$. If $a_{23}^{(3)} \neq 0$, then $a_{11}^{(3)} = a_{44}^{(3)} = 0$, $A^{(3)}$ is of the form

$$A^{(3)} = \begin{bmatrix}
0 & 0 & \times & 0 \\
0 & \times & \times & \\
\times & \times & \times & 0 \\
\times & \times & \times & 0
\end{bmatrix}.$$
Use a real symplectic double rotation $Q = \begin{bmatrix} c & 0 & 0 & s \\ 0 & c & s & 0 \\ 0 & -s & c & 0 \\ -s & 0 & 0 & c \end{bmatrix}$ to zero the $(2, 3)$ and $(3, 2)$ entries of $A^{(4)} = Q^{(3)^T} A^{(3)} Q^{(3)}$.

\[
\begin{bmatrix}
0 & 0 & \times & 0 \\
0 & \times & \times & \times \\
\times & \times & \times & 0 \\
0 & \times & 0 & 0
\end{bmatrix} \rightarrow \begin{bmatrix}
0 & 0 & \times & 0 \\
0 & \times & 0 & \times \\
\times & \times & \times & 0 \\
0 & \times & 0 & 0
\end{bmatrix}
\]

$A^{(3)}$ $A^{(4)}$

**Case 3.** $a_{14}^{(3)} = 0$ and $A^{(3)} = \begin{bmatrix} 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \\ \times & \times & \times & 0 \\ 0 & \times & 0 & 0 \end{bmatrix}$

Using $a_{12}^{(3)} = a_{14}^{(3)} = a_{34}^{(3)} = 0$, equations (4.4) and (4.5) give $a_{11}^{(3)} a_{23}^{(3)} = a_{23}^{(3)} a_{44}^{(3)} = 0$. If $a_{23}^{(3)} \neq 0$, then $a_{11}^{(3)} = a_{44}^{(3)} = 0$, $A^{(3)}$ is of the form

\[
A^{(3)} = \begin{bmatrix}
0 & 0 & \times & 0 \\
0 & 0 & \times & \times \\
\times & \times & \times & 0 \\
0 & \times & 0 & 0
\end{bmatrix}
\]

Use a real symplectic double rotation $Q = \begin{bmatrix} c & 0 & 0 & s \\ 0 & c & s & 0 \\ 0 & -s & c & 0 \\ -s & 0 & 0 & c \end{bmatrix}$ to zero the $(2, 3)$ and $(3, 2)$ entries of $A^{(4)} = Q^{(3)^T} A^{(3)} Q^{(3)}$.

\[
\begin{bmatrix}
0 & 0 & \times & 0 \\
0 & \times & \times & \times \\
\times & \times & \times & 0 \\
0 & \times & 0 & 0
\end{bmatrix} \rightarrow \begin{bmatrix}
0 & 0 & \times & 0 \\
0 & \times & 0 & \times \\
\times & \times & \times & 0 \\
0 & \times & 0 & 0
\end{bmatrix}
\]

$A^{(3)}$ $A^{(4)}$

**Step 5.** Use a real symplectic plane rotation $Q = \begin{bmatrix} c & 0 & 0 & s \\ 0 & 0 & s & 0 \\ 0 & -s & 0 & 0 \\ -s & 0 & 0 & 0 \end{bmatrix}$ to zero the $(1, 3)$ and $(3, 1)$ entries of $A^{(5)} = Q^{(4)^T} A^{(4)} Q^{(4)}$. As was discussed before, this similarity will affect only rows and columns 1,3 of $A^{(4)}$, and move all weight of the pivots into the diagonal. Moreover, it also preserves the norm of row pairs and column pairs, that is,

\[
(a_{1i}^{(5)})^2 + (a_{3i}^{(5)})^2 = (a_{1i}^{(4)})^2 + (a_{3i}^{(4)})^2 = 0, \quad i = 2, 4, \quad (4.18)
\]

\[
(a_{1i}^{(5)})^2 + (a_{3i}^{(5)})^2 = (a_{1i}^{(4)})^2 + (a_{3i}^{(4)})^2 = 0, \quad i = 2, 4. \quad (4.19)
\]

But $a_{1i}^{(4)} = a_{3i}^{(4)} = 0$ for $i = 2, 4$, from (4.18), we have $a_{1i}^{(5)} = a_{3i}^{(5)} = 0$, for $i = 2, 4$. Similarly, since $a_{1i}^{(4)} = a_{3i}^{(4)} = 0$ for $i = 2, 4$, from (4.19), we get
\[ a_{i1}^{(5)} = a_{i3}^{(5)} = 0, \text{ for } i = 2, 4. \]

\[
\begin{bmatrix}
\times & 0 & \diamond & 0 \\
0 & \times & 0 & \times \\
\diamond & 0 & \times & 0 \\
0 & \times & 0 & \times \\
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\times & 0 & 0 & 0 \\
0 & \times & 0 & \times \\
0 & 0 & \times & 0 \\
0 & 0 & \times & 0 \\
\end{bmatrix}
\]

\[ A^{(4)} \rightarrow A^{(5)} \]

**Step 6.** Finally, use a real symplectic plane rotation \( Q^{(5)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c & 0 & s \\ 0 & 0 & 1 & 0 \\ -s & 0 & 0 & c \end{bmatrix} \) to zero the (2, 4) and (4, 2) entries of \( A^{(6)} = Q^{(5)^T} A^{(6)} Q^{(5)} \). Once again, the resulting similarity affects only rows and columns 2, 4 of \( A^{(6)} \). This leads to the equations

\[
(a_{2i}^{(6)})^2 + (a_{4i}^{(6)})^2 = (a_{2i}^{(5)})^2 + (a_{4i}^{(5)})^2 = 0, \quad i = 1, 3, \quad (4.20)
\]

\[
(a_{i2}^{(6)})^2 + (a_{i4}^{(6)})^2 = (a_{i2}^{(6)})^2 + (a_{i4}^{(5)})^2 = 0, \quad i = 1, 3. \quad (4.21)
\]

That is, for \( i = 1, 3 \) we have \( a_{2i}^{(6)} = a_{4i}^{(6)} = 0 \) and \( a_{i2}^{(6)} = a_{i4}^{(6)} = 0 \) from (4.20) and (4.21), respectively. Note that this similarity will not destroy the zero entries in the positions (1, 3) and (3, 1) of \( A^{(6)} \) since they are in unaffected rows and columns.

\[
\begin{bmatrix}
\times & 0 & 0 & 0 \\
0 & \times & 0 & \diamond \\
0 & 0 & \times & 0 \\
\diamond & 0 & \times & 0 \\
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\times & 0 & 0 & 0 \\
0 & \times & 0 & 0 \\
0 & 0 & \times & 0 \\
0 & 0 & \times & 0 \\
\end{bmatrix}
\]

\[ A^{(5)} \rightarrow A^{(6)} \]

Upon completion, the real symplectic orthogonal similarity transformation relating the original matrix \( A \) to the final diagonal matrix \( A^{(6)} \) is

\[ Q = Q^{(0)} Q^{(1)} Q^{(2)} Q^{(3)} Q^{(4)} Q^{(5)}, \]

which is also a real symplectic orthogonal matrix (by closure under multiplication). Hence, a structured spectral decomposition of \( A \) can be written as

\[ A = Q^T A^{(6)} Q, \]

where \( A^{(6)} \) is diagonal.
4.2 Sweep Designs for $2n \times 2n$ Case

Let $A$ be a real $2n \times 2n$ symplectic symmetric matrix. Structure preserving algorithms for computing a structured spectral decomposition of $A$ can be developed once we determine how to design a "sweep" for our algorithms. Informally speaking, by a sweep we mean a pattern for choosing a sequence of pivot elements; this pattern is repeated till the matrix converges to the desired form.

Once a pivot has been chosen, then depending on its position, a real symplectic orthogonal transformation (or a tool) is constructed using an appropriate algorithm from Chapter 2. It will be convenient to write $A$ in block form, i.e.

$$A = \begin{bmatrix} E & F \\ F^T & H \end{bmatrix}, \quad E, F, H \in \mathbb{R}^{n \times n}, E^T = E, H^T = H.$$

- If a pivot element is an off-diagonal element of $E$ or $H$, then there is only one tool described in Chapter 2 that is used to annihilate this pivot by performing a Jacobi-like action — a real symplectic double rotation with direct sum embedding as described in Section 2.3.1.

- If a pivot element is a diagonal element of $F$ or $F^T$, then a real symplectic plane rotation is used to zero out this pivot by performing a Jacobi-like action as described in Section 2.2.2.

- If a pivot element is an off-diagonal element of $F$ or $F^T$, then this pivot is annihilated either by using a real symplectic plane rotation to perform a Givens-like action as described in Section 2.2.1 or by using a real symplectic double rotation with concentric embedding to perform a Jacobi-like action as described in Section 2.3.2.

Each iteration during a sweep corresponds to the similarity transformation

$$A^{(\ell+1)} = Q^{(\ell)} A^{(\ell)} Q^{(\ell)}^{-1},$$

where $Q^{(\ell)} \in SpO(2n, \mathbb{R})$ and $A^{(0)} = A$.

We have six different sweep designs for our algorithms, called strategies $S_1, S_2, S_3, S_4, S_5,$ and $S_6$. Our aim for all designs is to zero out the off-diagonal entries of $A$, that is, move all the "weight" of off-diagonal entries entirely onto the main diagonal of $A$. As we see in Section 4.3, numerical experiments presented show that the first three strategies, $S_1, S_2,$ and $S_3,$ require a high number of sweeps for convergence, and are too expensive for practical use. We present them here for the record nonetheless, in the hope that this will forestall reinvestigation by other explorers.
4.2.1 Strategy $S_1$

The strategy here is to first reduce the matrix $A = \begin{bmatrix} E & F \\ F^T & H \end{bmatrix}$ to a matrix $\tilde{A}$ with diagonal blocks, that is, $\tilde{A} = \begin{bmatrix} \begin{bmatrix} \vdots \end{bmatrix} & \begin{bmatrix} \vdots \end{bmatrix} \\ \begin{bmatrix} \vdots \end{bmatrix} & \begin{bmatrix} \vdots \end{bmatrix} \end{bmatrix}$, and then reduce $\tilde{A}$ to diagonal form by using a finite sequence of real symplectic orthogonal similarities. We illustrate the sweep design for the reduction of $A$ to $\tilde{A}$ in Figure 4.1; the finite steps of the reduction of $A$ to diagonal form are illustrated in Figure 4.2, when $A$ is a general (not necessarily symplectic) $6 \times 6$ real symmetric matrix. The two entries represented by $\diamond$ determine the pivots of the transformation (one of these by symmetry), and the entries denoted by $\bullet$ determine where either one or two copies of a plane rotation $\begin{bmatrix} c & s \\ -s & c \end{bmatrix}$ are embedded as a principal submatrix of $I_6$ to give a real symplectic orthogonal matrix used in a similarity transformation.

In order to reduce $A$ to $\tilde{A}$, let us first consider the sequence of pivot elements, which, for this design, can be grouped in two sets:

Set 1. Pivot elements are chosen in row-cyclic order in the strict upper triangular part of $F$ interleaved with pivot elements in the strict upper triangular part of $E$.

Set 2. Pivot elements are chosen in column-cyclic order in the strict lower triangular part of $F$ interleaved with pivot elements in the strict lower triangular part of $H$.

Next we consider tools used to annihilate the pivots in this design.

- If a pivot element is in the strict upper triangular part of $E$ or in the strict lower triangular part of $H$, then it is set to zero by a Jacobi-like action using a real symplectic double plane rotation with direct sum embedding as described in Section 2.3.1.

- If a pivot element is in the strict upper triangular part of $F$, then it is made zero by a Givens-like right action using a real symplectic plane rotation as described in Section 2.2.1. On the other hand, if a pivot element is in the strict lower triangular part of $F$, then it is set zero by a Givens-like left action as described in Section 2.2.1.

Let us now consider the action of the first two iterations from Figure 4.1, displayed in (4.22) and (4.23), respectively. Here the entry denoted by $\Delta$ is a partner of the pivot element during transformation; the entry denoted $\triangle$ carries all weight of the pivot element.

As shown in (4.22), we first target the $(1, 5)$ entry in the strict upper triangular
Figure 4.1: Sweep design for Strategy $S_1$ to reduce a real symplectic symmetric matrix to a matrix with diagonal blocks

part of $F$; this entry is made zero by a Givens-like right action using a real symplectic plane rotation $Q$ as described in Section 2.2.1. This action pairs the $(1,5)$ and $(1,2)$ entries of $A$, keeping the quantity $|a_{15}|^2 + |a_{12}|^2$ invariant. But the $(1,5)$ entry is annihilated; its weight is therefore transferred onto the $(1,2)$ entry, which consequently grows in absolute value. Moreover, as discussed in Section 2.2.1, this left action affects only columns 2 and 5 of $A$ and by using (2.24) we have $|(AQ)_{22}|^2 + |(AQ)_{25}|^2 = |a_{22}|^2 + |a_{25}|^2$ and $|(AQ)_{52}|^2 + |(AQ)_{56}|^2 = |a_{52}|^2 + |a_{56}|^2$. This implies that there is liable to be some leakage of the weight from the main diagonal entries $(2,2)$ and $(5,5)$ into the off-diagonal entries $(2,5)$ and $(5,2)$.

Next, in order to preserve the symmetry of $A$ and complete the similarity, left-multiplication on $AQ$ by $Q^T$ is needed, and hence we have the similarity $A^{(1)} = Q^TAQ$. By this left action the $(5,1)$ entry is also made zero, and the $(2,1)$ entry also grows in absolute value. Again, this action affects only rows 2 and 5 of $AQ$ and by using (2.21) we obtain $|a_{22}^{(1)}|^2 + |a_{52}^{(1)}|^2 = |(AQ)_{22}|^2 + |(AQ)_{52}|^2$ and
Thus, there is again liable to be some leakage of the weight from the main diagonal entries (2,2) and (5,5) into the off-diagonal entries (5,2) and (2,5).

The action of the second iteration is shown in (4.23). In this iteration, we directly target the (1,2) and (2,1) entries which have just increased in absolute value, and move their weight onto the diagonal of \( E \) using a similarity by \( Q \), which is now a real symplectic double plane rotation with direct sum embedding as described in Section 2.3.1. The similarity transformation \( A^{(2)} = Q^T A^{(1)} Q \) will affect rows and columns 1, 2, 4, and 5 of \( A^{(1)} \); the (1,5) and (5,1) entries annihilated in the previous iteration gain some weight since they lie in the affected rows and columns.

Moreover, as discussed in Section 2.3.1, there is liable to be some leakage of the weight from the diagonal entries (4,4) and (5,5) in the lower diagonal block into the off-diagonal entries (4,5) and (5,4), that is, into two of the off-diagonal elements that we hope to zero out eventually.

\[
\begin{align*}
\frac{\Delta}{\bullet} & \quad \bullet \quad \bullet \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\
\cdot & \quad \bullet \quad \Delta \quad \bullet \quad \cdot \quad \cdot \quad \cdot \\
\cdot & \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\
\cdot & \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\
\end{align*}
\]

Note that the first two iterations just described were part of Set 1. We now consider the action of two more iterations, iterations 7 and 8 from Figure 4.1, which are part of Set 2. We display the action of these two iterations in (4.24) and (4.25), respectively. First, we consider the action of iteration 7. As shown in (4.24), a pivot, the (2,4) entry, is in the strict lower triangular part of \( F \); it is made zero by a Givens-like left action using a real symplectic plane rotation \( Q^T \) as described in Section 2.2.1. Its weight moves onto the (5,4) entry. The similarity is completed by right-multiplication on \( Q^T A \) by \( Q \). By this right action, the (4,2) entry is made zero, and its weight is transferred onto the (4,5) entry.

\[
\begin{align*}
\frac{\bullet}{\Delta} & \quad \bullet \quad \bullet \quad \cdot \quad \cdot \quad \cdot \\
\frac{\bullet}{\Delta} & \quad \bullet \quad \Delta \quad \bullet \quad \cdot \quad \cdot \\
\cdot & \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\
\cdot & \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\
\cdot & \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\
\cdot & \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\
\end{align*}
\]

\[
\begin{align*}
\frac{\Delta}{\bullet} & \quad \bullet \quad \bullet \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\
\cdot & \quad \bullet \quad \Delta \quad \bullet \quad \cdot \quad \cdot \quad \cdot \\
\cdot & \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\
\cdot & \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\
\cdot & \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\
\cdot & \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\
\end{align*}
\]
Next, let consider the action of iteration 8 as shown in (4.25). In this iteration, weight of the pivots, the (4,5) and (5,4) entries, which have increased in absolute value as a consequence of iteration 7, moves onto the diagonal of \( H \) by a Jacobi-like action using a real symplectic double rotation with direct sum embedding \( Q \) as described in Section 2.3.1. Again, there is liable to be some leakage of the weight from the diagonal entries (1,1) and (2,2) in the upper diagonal block into the off-diagonal entries (1,2) and (2,1).

\[
\begin{bmatrix}
\vdots & \vdots & \vdots & \ddots & \vdots \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\end{bmatrix} 
\Rightarrow 
\begin{bmatrix}
\vdots & \vdots & \vdots & \ddots & \vdots \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\end{bmatrix}
\]

(4.25)

Subsequent iterations of this design sweep are depicted analogously in Figure 4.1. Note that in this design, the diagonals of \( E \) and \( H \) are not a safe haven for the norm of \( A \). Safe haven means a place from which weight never gets expelled by any iteration. As we discussed above, at every iteration, some weight from the diagonal entries of \( E \) and \( H \) could move into the off-diagonal entries of \( A \). In (4.23) weight moves from diagonal of \( H \) to off-diagonal of \( H \), while in (4.25) weight moves from diagonal of \( E \) to off-diagonal of \( E \). This is confirmed by numerical experiments.

Our sweep design for this strategy is represented below as a sequence of pivot elements for simplicity:

Set 1: \((1, n + 2), (1, 2), (1, n + 3), (1, 3), (1, n + 4), (1, 4), \ldots, (1, 2n), (1, n), (2, n + 3), (2, 3), (2, n + 4), (2, 4), \ldots, (2, 2n), (2, n), (3, n + 4), (3, 4), \ldots, (3, 2n), (3, n), \ldots, (n - 1, 2n), (n - 1, n),\)

Set 2: \((2, n + 1), (n + 1, n + 2), (3, n + 1), (n + 1, n + 3), \ldots, (n, n + 1), (n + 1, 2n), (3, n + 2), (n + 2, n + 3), \ldots, (n, n + 2), (n + 2, 2n), \ldots, (n, 2n - 1), (2n - 1, 2n)\)

Next, we examine Figure 4.2 to understand why the reduction of \( \tilde{A} \) to diagonal form can be accomplished in finite steps. For example, we consider the first iteration in Figure 4.2. Zeroing out the off-diagonal entries (1,4) and (4,1) by performing a Jacobi-like action using a plane rotation as in Section 2.2.2 moves all weight of pivot elements into the diagonal entries (1,1) and (4,4), and no weight leaks out from any diagonal entry, as discussed in Section 2.2.2. Thus when \( n = 3 \), the reduction of \( \tilde{A} \) to diagonal form can be achieved by performing...
at most 3 Jacobi-like iterations, one for each non-zero diagonal entry of $F$.

\[
\begin{bmatrix}
& \star & \star & \star \\
& \star & \star & \star \\
& \star & \star & \star \\
& \star & \star & \star \\
\end{bmatrix}
\rightarrow
\begin{bmatrix}
& \star & \star & \star \\
& \star & \star & \star \\
& \star & \star & \star \\
& \star & \star & \star \\
\end{bmatrix}
\rightarrow
\begin{bmatrix}
& \star & \star & \star \\
& \star & \star & \star \\
& \star & \star & \star \\
& \star & \star & \star \\
\end{bmatrix}
\]

Figure 4.2: Reduction of a diagonal block matrix to diagonal form

In this strategy, the number of iterations per sweep for the reduction of $A$ to

\[
\tilde{A} = \begin{bmatrix}
& & & \\
& & & \\
& & & \\
& & & \\
\end{bmatrix}
\]

is at most $2n^2 - 2n$. An additional $n$ iterations are needed for
the reduction of $\tilde{A}$ to diagonal form.

4.2.2 Strategy $S_2$

Looking back to the sweep design for strategy $S_1$, we see that $A = \begin{bmatrix} E & F \\ F^T & H \end{bmatrix}$ was
diagonalized by first reducing to a diagonal block matrix $\tilde{A} = \begin{bmatrix}
& & & \\
& & & \\
& & & \\
& & & \\
\end{bmatrix}$. This brings us to the question of whether a real symplectic symmetric matrix $A$ can be
diagonalized without the initial reduction to $\tilde{A}$. Trying to answer this question
gives us an idea to explore a new sweep design, which we call $S_2$. We illustrate
this sweep design in Figure 4.3. In this design the sequence of pivot elements can
be grouped as follows:

Set 1. Pivot elements are chosen in row-cyclic order from the upper triangular
part of $F$, interleaved with pivots chosen from the strict upper triangular
part of $E$.

Tools:
Diagonal elements of $F$ are zeroed out by Jacobi-like actions via symplectic
plane rotations as in Section 2.2.2. These transformations move weight from
the diagonal elements of $F$ onto the diagonal elements of $E$.
Off-diagonal elements in the upper triangular part of $F$ are zeroed out by
Givens-like right actions performed using symplectic plane rotations as in
Section 2.2.1. Every such iteration moves the weight of the target into an
off-diagonal element of $E$. This element of $E$ becomes the target of the next
iteration, which uses a double rotation as in Section 2.3.1 with the rotation
in the upper diagonal block being the active one.
Set 2. Pivot elements are chosen in column-cyclic order from the strict lower triangular part of $F$ interleaved with pivots chosen from the strict lower triangular part of $H$.

Tools:
This time, the off-diagonal elements of $F$ are zeroed out by Givens-like left actions. Every such iteration is followed by a double rotation with the rotation in the lower diagonal block being the active one.

The details of each iteration in this design can be described analogously as in Section 4.2.1.

Figure 4.3: Sweep design for strategy $S_2$
In general, the sequence of pivot elements can be written as follows.

Set 1: 
\begin{align*}
(1, n + 1), (1, n + 2), (1, 2), (1, n + 3), (1, 3), \ldots, (1, 2n), (1, n), \\
(2, n + 2), (2, n + 3), (2, 3), \ldots, (2, 2n), (2, n), \\
\vdots \\
(n, 2n),
\end{align*}

Set 2: 
\begin{align*}
(2, n + 1), (n + 1, n + 2), (3, n + 1), (n + 1, n + 3), \ldots, (n, n + 1), (n + 1, 2n), \\
(3, n + 2), (n + 2, n + 3), \ldots, (n, n + 2), (n + 2, 2n), \\
\vdots \\
(n, 2n - 1), (2n - 1, 2n)
\end{align*}

Note that the number of iterations for each sweep in this design is at most \(2n^2 - n\).

### 4.2.3 Strategy \(S_3\)

The sweep design for strategy \(S_3\) is obtained by reordering the sequence of pivot elements of strategy \(S_2\). Rather than dividing the elements of \(F\) into two sets — those in the upper triangular part (Set 1) and those in the strict lower triangular part (Set 2), this new strategy simply targets the elements of the entire matrix \(F\) in row-cyclic order. As before, every iteration that targets an off-diagonal entry of \(F\) is followed by one that targets an appropriate of diagonal entry of \(E\) or \(H\). The tools and actions are the same as those used in strategy \(S_2\). Thus, each time an off-diagonal entry of \(F\) is zeroed out by a Givens-like right action or a Givens-like left action, then the pivot element of the following rotation is chosen in the strict upper triangular part of \(E\) or the strict upper triangular part of \(H\) and is zeroed out by a real symplectic double rotation. Thus, the number of iterations per sweep in this design is the same as that of strategy \(S_2\). We illustrate this sweep design in Figure 4.4.

The description of iterations in this design can be described analogously as in Section 4.2.1.
In general this sweep design can be expressed as the sequence of pivot elements in the following.

\[(1, n + 1), (1, n + 2), (1, 2), (1, n + 3), (1, 3), \ldots, (1, 2n), (1, n),
\]
\[(2, n + 1), (n + 1, n + 2), (2, n + 2), (2, n + 3), (2, 3), \ldots, (2, 2n), (2, n),
\]
\[(3, n + 1), (n + 1, n + 3), (3, n + 2), (n + 2, n + 3), \ldots, (3, 2n), (n + 1, 2n),
\]
\[(3, n + 2), (n + 2, n + 3), \ldots, (n, n + 2), (n + 2, 2n),
\]
\[\vdots
\]
\[(n, 2n - 1), (2n - 1, 2n)
\]

Note that the number of iterations per sweep in this design is at most \(2n^2 - n\).
4.2.4 Strategy $S_4$

The sweep design for strategy $S_4$ is inspired from the $4 \times 4$ case described in Section 4.1, that is, we use a combination of double rotations and plane rotation to perform only Jacobi-like actions, avoiding Givens-like action entirely. In this design, the sequence of pivot elements can be grouped into three sets. Recall $A = [\begin{array}{cc} E & F \\ F^T & H \end{array}]$.

**Set 1.** Pivot elements are diagonal entries of $F$ and chosen in following order: $(1, n + 1), (2, n + 2), \ldots, (n, 2n)$. Pivots are annihilated by a Jacobi-like action using real symplectic plane rotations as described in Section 2.2.2.

**Set 2.** Pivot elements are chosen in row-cyclic order in the strict upper triangular part of $E$; they are made zero by a Jacobi-like action using real symplectic double rotations with direct sum embedding as described in Section 2.3.1.

**Set 3.** Pivot elements are chosen in row-cyclic order in the strict upper triangular part of $F$; they are made zero by a Jacobi-like action using real symplectic double rotations with concentric embedding as described in Section 2.3.2.

We illustrate this sweep design in Figure 4.5.

![Figure 4.5: Structured sweep for Strategy $S_4$](image)

It can be seen from Figure 4.5 that the first three iterations make the diagonal of $F$ zero by Jacobi-like actions using real symplectic plane rotations; all the
weight of the pivots is transferred onto the main diagonal of $E$, and hence onto the main diagonal of $A$. There is no leakage of weight from any of the main diagonal elements by doing these actions.

In the next three iterations, we target the elements in the strict upper triangular part of $E$; these pivots are made zero by Jacobi-like actions using real symplectic double rotations with direct sum embedding. The active rotations move all the weight of the pivots onto the diagonal entries of $E$. But the weight of the affected diagonal entries in the corresponding positions in block $H$ moves around, due to the action of the inactive rotation. That is, there is liable to be some leakage of the weight from the diagonal entries of $H$ into the off-diagonal entries of $H$.

In the last three iterations in Figure 4.5, the similarities are performed using real symplectic double rotations with concentric embedding as described in Section 2.3.2. In each of these iterations, the pivot elements are made zero by the outer rotation. As discussed in Section 2.3.2, there is liable to be leakage of weight from the diagonal entries of $E$ and $H$ to the off-diagonal part of $F$ and $F^T$ by the action of inner rotation.

This sweep design can be represented as the following sequence of pivot elements.

$$(1, n + 1), (2, n + 2), (3, n + 3), \ldots, (n, 2n),$$

$$(1, 2), (1, 3), (1, 4), \ldots, (1, n),$$

$$(2, 3), (2, 4), \ldots, (2, n),$$

$$(3, 4), \ldots, (3, n),$$

$$\vdots$$

$$(n - 1, n),$$

$$(1, n + 2), (1, n + 3), (1, n + 4), \ldots, (1, 2n),$$

$$(2, n + 3), (2, n + 4), \ldots, (2, 2n),$$

$$(3, n + 4), \ldots, (2, 2n),$$

$$\vdots$$

$$(n - 1, 2n)$$

Note that most $n^2$ iterations per sweep are needed by this design. Let subscript $s$ denote the use of the sorting angle. The following gives the description of the use of the sorting angle for this design, called strategy $S_{4s}$.

**Sorting Version $S_{4s}$**: Because our algorithms are structure preserving, the computed diagonal matrix is symplectic and hence of the form $\Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n, \frac{1}{\sigma_1}, \frac{1}{\sigma_2}, \ldots, \frac{1}{\sigma_n})$. When the small angle is used at every iteration, the entries $\sigma_1, \sigma_2, \ldots, \sigma_n$ do not appear in any particular order on the diagonal. In order to have these values appear in decreasing order, the sorting angle needs to be introduced. Observe that the pivot elements for this design occur only in the
first $n$ rows of $A$. By using the sorting angle at every iteration of every sweep, strategy $S_{4s}$ results in the first $n$ diagonal entries $\sigma_1, \sigma_2, \ldots, \sigma_n$ of $\Sigma$ appearing, at convergence, in decreasing order on the diagonal. The remaining $n$ diagonal entries of $\Sigma$, being reciprocally paired with the first $n$ entries, therefore appear in increasing order. We point out that, in practice, $\sigma_1, \sigma_2, \ldots, \sigma_n$ are the $n$ largest singular values from among the complete set of $2n$ singular values of $A$. We believe that this occurs because the use of the sorting angle for a real symplectic plane rotation and a real symplectic double rotation with concentric embedding brings the larger diagonal entry in the diagonal of $H$ up to the diagonal of $E$. At the same time, using the sorting angle for a real symplectic double rotation with direct sum embedding with the active rotation in upper diagonal block sorts the diagonal entries of $E$.

4.2.5 Strategy $S_5$

The sequence of pivot elements for strategy $S_5$ is designed in row-by-row fashion. That is, for a $6 \times 6$ matrix $A$ we cycle as follows:

$(1,2), (1,3), (1,4), (1,5), (1,6),$
$(2,3), (2,4), (2,5), (2,6),$
$(3,4), (3,5), (3,6),$
$(4,5), (4,6),$
$(5,6),$

(1,2), \ldots$

The sweep design of this strategy is shown in Figure 4.6. Tools used to annihilate the pivots in this design are as follows. Recall $A = [E \ F \ H]$:

- If a pivot is in the strict upper triangular part of $E$ or $H$, then it is made zero by a Jacobi-like action using a real symplectic double rotation with direct sum embedding as described in Section 2.3.1.

- If a pivot is a diagonal entry of $F$, then it is annihilated by a Jacobi-like action using a real symplectic plane rotation as described in Section 2.2.2.

- If a pivot is in the strict upper triangular part of $F$, then it is made zero by the outer rotation of a real symplectic double rotation with concentric embedding as described in Section 2.3.2.

- If a pivot is in the strict lower triangular part of $F$, then the inner rotation of a real symplectic double rotation as described in Section 2.3.2 is the one that makes this pivot zero.
Each iteration of this design can be described analogously as in the previous sections. That is, if a pivot element is annihilated by a Jacobi-like action using a real symplectic plane rotation as in Section 2.2.2, then all the weight of the pivot is transferred onto the main diagonal, and there is no leakage of weight from any of the main diagonal elements. However, whenever a pivot element is made zero by a Jacobi-like action using a real symplectic double rotation, then the active plane rotation moves the annihilated weight onto the diagonal of $E$ or $H$ while the other inactive rotation may cause weight to move out from diagonal entries.

Figure 4.6: Structured sweep for Strategy $S_6$
to off-diagonal entries. When this happens, the diagonal is not a safe haven for the norm of $A$.

In general the sequence of the pivot elements of this design can be represented in the following.

$$(1, 2), (1, 3), (1, 4), \ldots, (1, 2n),$$

$$ (2, 3), (2, 4), \ldots, (2, 2n),$$

$$ (3, 4), \ldots, (3, 2n),$$

$$ \vdots $$

$$(2n - 1, 2n)$$

Note that the number of iterations per sweep of this design is at most $2n^2 - n$.

**Sorting Version $S_{5s}$**: By contrast to the design $S_4$, the pivot elements in strategy $S_5$ are not limited to the first $n$ rows of $A$. If the sorting angle is used at every iteration during every sweep of $S_5$, the resulting algorithm does not converge. The reason is that the symplectic structure forges a connection between the upper and lower diagonal block of $\Sigma$. Indiscriminate use of the sorting angle causes repeated re-orderings which are counter productive and lead to oscillations and non-convergence. We therefore restrict the use of the sorting angle as follows in strategy $S_{5s}$:

- If a pivot element is in the strict upper triangular part of $E$ or the block submatrix $F$, then the sorting angle for a rotation is used.

- If a pivot element is in the strict upper triangular part of $H$, then the small angle for a rotation is used. This preserves the order of the affected diagonal elements.

Once again, in practice, the first $n$ diagonal entries of $\Sigma$ are the $n$ largest singular values of $A$. We believe that the explanation given in the previous section also applies here.

**4.2.6 Strategy $S_6$**

The sweep design for strategy $S_6$ is obtained by reordering the sequence of pivot elements of strategy $S_5$. The sequence of pivot elements can be grouped into five sets. Recall $A = \begin{bmatrix} E & F \\ F^T & H \end{bmatrix}$.

**Set 1.** Pivot elements are chosen in row-cyclic order in the strict upper triangular part of $E$. Pivots are annihilated by a Jacobi-like action using real symplectic double rotations with direct sum embedding as described in Section 2.3.1 with the rotation in the upper diagonal block being the active one.
Set 2. Pivot elements are diagonal entries of $F$, chosen in the following order: $(1, n + 1)$, $(2, n + 2)$, ..., $(n, 2n)$. Pivots are made zero by a Jacobi-like action using real symplectic plane rotations as described in Section 2.2.2.

Set 3. Pivot elements are chosen in row-cyclic order in the strict upper triangular part of $F$; they are made zero by the outer rotation of real symplectic double rotations with concentric embedding as described in Section 2.3.2.

Set 4. Pivot elements are chosen in column-cyclic order in the strict lower triangular part of block $F$; they are annihilated by the inner rotation of real symplectic double rotations with concentric embedding as described in Section 2.3.2.

Set 5. Pivot elements are chosen in row-cyclic order in the strict upper triangular part of $H$; they are made zero by a Jacobi-like action using real symplectic double rotations with direct sum embedding as described in Section 2.3.1 with the rotation in the lower diagonal block being the active one.

We illustrate this design in Figure 4.7. The description of each iteration in this design is analogous to that given in Section 4.2.5.
The sequence of pivot elements of this design can be represented as follows.

Set 1: \((1, 2), (1, 3), (1, 4), \ldots, (1, n),\)
\((2, 3), (2, 4), \ldots, (2, n),\)
\((3, 4), \ldots, (3, n),\)
\(\vdots\)
\((n - 1, n),\)

Set 2: \((1, n + 1), (2, n + 2), (3, n + 3), \ldots, (n, 2n),\)

Set 3: \((1, n + 2), (1, n + 3), (1, n + 4), \ldots, (1, 2n),\)
\((2, n + 3), (2, n + 4), \ldots, (2, 2n),\)
\((3, n + 4), \ldots, (3, 2n),\)
\(\vdots\)
\((n - 1, 2n),\)

Set 4: \((2, n + 1), (3, n + 1), (4, n + 1), \ldots, (n, n + 1),\)
\((3, n + 2), (4, n + 2), \ldots, (n, n + 2),\)
\((4, n + 3), \ldots, (n, n + 3),\)
\(\vdots\)
\((n, 2n - 1),\)

Set 5: \((n + 1, n + 2), (n + 1, n + 3), (n + 1, n + 4), \ldots, (n + 1, 2n),\)
\((n + 2, n + 3), (n + 2, n + 4), \ldots, (n + 2, 2n),\)
\((n + 3, n + 4), \ldots, (n + 3, 2n),\)
\(\vdots\)
\((2n - 1, 2n)\)

Note that the number of iterations for each sweep in this design is at most \(2n^2 - n\).

**Sorting Version \(S_6s\):** Once again, if the sorting angle is used at every iteration, numerical experiments show that our algorithm with this design does not converge. To have the first \(n\) diagonal entries of \(\Sigma\) appear in decreasing order,
the sorting angle is used when the pivot element belongs in the Sets 1-4 of this design. This strategy is analogous to the one used in \( S_5 \).

Our design for the use of the sorting angle is motivated by our knowledge the symplectic structure of \( \Sigma \) which is being preserved will force the values and the order of the remaining \( n \) diagonal entries, which have to be reciprocals of the first \( n \) singular values.

Note that the sweep designs \( S_4, S_5 \) and \( S_6 \) are the main designs for all our algorithms. Observe that, in these designs, every pivot element is the target element exactly once during a sweep and this pattern is repeated until algorithm converges.

### 4.3 Description of the Algorithms

In this section we present structure perserving algorithms for computing a spectral decomposition of a given real \( 2n \times 2n \) symplectic symmetric matrix \( A \) by using all sweep designs in previous section. Define \( \text{reloff}(A) = \text{off}(A)/\|A\|_F \) where \( \text{off}(A) \) is defined as in (2.15) for strategies \( S_2, S_3, S_4, S_5, \) and \( S_6 \), while \( \text{off}(A) \) is the off-diagonal norm of block submatrices \( E, H, F, F^T \) for strategy \( S_1 \). \( \|A\|_F \) is the Frobenius norm of \( A \). The goal of all algorithms is to make \( \text{reloff}(A) \) approach zero. Here the stopping criterion for \( \text{reloff}(A) \) is denoted by \( \text{reloff}.\text{tol} \). That is, the process of all algorithms is terminated when \( \text{reloff}(A) \leq \text{reloff}.\text{tol} \).

**Algorithm 4 (Strategy \( S_1 \)).** Given a symplectic symmetric matrix \( A \in \mathbb{R}^{2n \times 2n} \), a Boolean parameter "sort", \( \text{tol} > 0 \), and \( \text{reloff}.\text{tol} > 0 \), this algorithm overwrites \( A \) with a diagonal matrix \( R^T AR \) where \( R \) is real symplectic orthogonal and \( \text{reloff}(R^T AR) \leq \text{reloff}.\text{tol} \).

\[
R = I_{2n}, \quad x_1 = \text{off}(E); \quad x_2 = \text{off}(H); \quad x_3 = \text{off}(F); \quad x_4 = \text{off}(F^T) \\
x = [x_1 \ x_2 \ x_3 \ x_4]; \quad \text{reloff}(A) = \frac{\|x\|_2}{\|A\|_F} \\
\text{while } \text{reloff}(A) > \text{reloff}.\text{tol} \\
\text{% Reduction of } A \text{ to } \tilde{A} = \begin{bmatrix} x_1 & \ldots & x_{2n} \end{bmatrix}. \\
\text{for } i = 1 : n - 1 \\
\quad \text{for } j = n + 1 + i : 2n \\
\quad \quad k = j - n \\
\quad \quad Q = \text{Givens.right}(A, i, j, \text{tol}) \quad \% \text{Section 2.2.1} \\
\quad \quad A = Q^T AQ; \quad R = RQ \\
\quad \quad Q = \text{Double}(A, i, k, \text{sort}, \text{tol}) \quad \% \text{Section 2.3.1} \\
\quad \quad A = Q^T AQ; \quad R = RQ
\]
Algorithm 5 (Strategy $S_2$). Given a symplectic symmetric matrix $A \in \mathbb{R}^{2n \times 2n}$, a Boolean parameter "sort", tol $> 0$, and reloff.tol $> 0$, this algorithm overwrites $A$ with a diagonal matrix $R^TAR$ where $R$ is real symplectic orthogonal and reloff($R^TAR$) $\leq$ reloff.tol.

$R = I_{2n}$

reloff($A$) = $\frac{\text{off}(A)}{\|A\|_F}$

while reloff($A$) $>$ reloff.tol

for $i = 1 : n$

for $j = n + 1 : 2n$

if $j = n + i$

$Q = \text{Rotation}(A, i, \text{sort}, \text{tol})$  \hspace{1cm} % Section 2.2.2

$A = Q^T AQ; \ R = RQ$

else

$k = j - n$

$Q = \text{Givens.right}(A, i, j, \text{tol})$  \hspace{1cm} % Section 2.2.1

$A = Q^T AQ; \ R = RQ$

$Q = \text{Double}(A, i, k, \text{sort}, \text{tol})$  \hspace{1cm} % Section 2.3.1

$A = Q^T AQ; \ R = RQ$

endif

endfor

endfor

$x_1 = \text{off}(E); \ x_2 = \text{off}(H); \ x_3 = \text{off}(F); \ x_4 = \text{off}(F^T)$

$x = [x_1 \ x_2 \ x_3 \ x_4]; \ \text{reloff}(A) = \frac{\|x\|_2}{\|A\|_F}$

endwhile

% Reduction to diagonal form.

for $i = 1 : n$

$Q = \text{Rotation}(A, i, \text{sort}, \text{tol})$  \hspace{1cm} % Section 2.2.2

$A = Q^T AQ; \ R = RQ$

endfor
endfor
for \( j = n + 1 : 2n - 1 \)
    for \( i = j - n + 1 : n \)
        \( k = i + n \)
        \( Q = \text{Givens.left}(A, i, j, tol) \) \hspace{1cm} \% Section 2.2.1
        \( A = Q^T AQ; R = RQ \)
        \( Q = \text{Double}(A, j, k, 0, tol) \) \hspace{1cm} \% Section 2.3.1
        \( A = Q^T AQ; R = RQ \)
    endfor
endfor
reloff(A) = \frac{\text{off}(A)}{\|A\|_F}
endwhile

Algorithm 6 (Strategy \( S_3 \)). Given a symplectic symmetric matrix \( A \in \mathbb{R}^{2n \times 2n} \), a Boolean parameter “sort”, \( tol > 0 \), and \( \text{reloff.tol} > 0 \), this algorithm overwrites \( A \) with a diagonal matrix \( R^T AR \) where \( R \) is real symplectic orthogonal and \( \text{reloff}(R^T AR) \leq \text{reloff.tol} \).

\( R = I_{2n} \)
\( \text{reloff}(A) = \frac{\text{off}(A)}{\|A\|_F} \)
while \( \text{reloff}(A) > \text{reloff.tol} \)
    for \( i = 1 : n \)
        for \( j = n + 1 : 2n \)
            if \( j = n + i \)
                \( Q = \text{Rotation}(A, i, \text{sort}, tol) \) \hspace{1cm} \% Section 2.2.2
                \( A = Q^T AQ; R = RQ \)
            elseif \( j > n + i \)
                \( k = j - n \)
                \( Q = \text{Givens.right}(A, i, j, tol) \) \hspace{1cm} \% Section 2.2.1
                \( A = Q^T AQ; R = RQ \)
                \( Q = \text{Double}(A, i, k, \text{sort}, tol) \) \hspace{1cm} \% Section 2.3.1
                \( A = Q^T AQ; R = RQ \)
            else
                \( k = i + n \)
                \( Q = \text{Givens.left}(A, i, j, tol) \) \hspace{1cm} \% Section 2.2.1
                \( A = Q^T AQ; R = RQ \)
                \( Q = \text{Double}(A, j, k, 0, tol) \) \hspace{1cm} \% Section 2.3.1
                \( A = Q^T AQ; R = RQ \)
            endif
        endfor
    endfor
endfor
\[
\text{reloff}(A) = \frac{\text{off}(A)}{\|A\|_F}
\]

\text{endwhile}

**Algorithm 7 (Strategy $S_4$, $S_{4s}$).** Given a symplectic symmetric matrix \( A \in \mathbb{R}^{2n \times 2n} \), a Boolean parameter "sort", \( \text{tol} > 0 \), and \( \text{reloff.tol} > 0 \), this algorithm overwrites \( A \) with a diagonal matrix \( \Sigma = R^T A R \) where \( R \) is real symplectic orthogonal and \( \text{reloff}(R^T A R) \leq \text{reloff.tol} \). If "sort" is zero, then the small angle is used at every iteration. On the other hand, if "sort" is nonzero, then the sorting angle is used at every iteration and results in the first \( n \) diagonal entries of \( \Sigma \) appearing in decreasing order.

\[ R = I_{2n} \]
\[ \text{reloff}(A) = \frac{\text{off}(A)}{\|A\|_F} \]

while \( \text{reloff}(A) > \text{reloff.tol} \)
\[ \text{for } i = 1 : n \]
\[ Q = \text{Rotation}(A, i, \text{sort}, \text{tol}) \quad \% \text{Section 2.2.2} \]
\[ A = Q^T AQ; R = RQ \]
\[ \text{endfor} \]
\[ \text{for } i = 1 : n - 1 \]
\[ \text{for } j = i + 1 : n \]
\[ Q = \text{Double}(A, i, j, \text{sort}, \text{tol}) \quad \% \text{Section 2.3.1} \]
\[ A = Q^T AQ; R = RQ \]
\[ \text{endfor} \]
\[ \text{endfor} \]
\[ \text{for } i = 1 : n - i \]
\[ \text{for } j = n + i + 1 : 2n \]
\[ Q = \text{Concentric}(A, i, j, \text{sort}, \text{tol}) \quad \% \text{Section 2.3.2} \]
\[ A = Q^T AQ; R = RQ \]
\[ \text{endfor} \]
\[ \text{endfor} \]
\[ \text{reloff}(A) = \frac{\text{off}(A)}{\|A\|_F} \]
\[ \text{endwhile} \]

**Algorithm 8 (Strategy $S_5$, $S_{5s}$).** Given a symplectic symmetric matrix \( A \in \mathbb{R}^{2n \times 2n} \), a Boolean parameter "sort", \( \text{tol} > 0 \), and \( \text{reloff.tol} > 0 \), this algorithm overwrites \( A \) with a diagonal matrix \( \Sigma = R^T A R \) where \( R \) is real symplectic orthogonal and \( \text{reloff}(R^T A R) \leq \text{reloff.tol} \). If "sort" is zero, then the small angle is used at every iteration. On the other hand, if "sort" is nonzero, then the sorting angle is used when the pivot element is in the first \( n \) rows, that is, row \( i = 1, 2, \ldots, n \), and results in the first \( n \) diagonal entries of \( \Sigma \) appearing in decreasing order.

69

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
\[
R = I_{2n}
\]
\[
\text{rel}(A) = \frac{\text{off}(A)}{\|A\|_F}
\]
while \( \text{rel}(A) > \text{rel}(\text{tol}) \)
  for \( i = 1 : n \)
    for \( j = i + 1 : 2n \)
      if \( j \leq n \)
        \[
        Q = \text{Double}(A, i, j, \text{sort}, \text{tol}) \]
        \[
        A = Q^T A Q; R = RQ
        \]
      elseif \( j = n + i \)
        \[
        Q = \text{Rotation}(A, i, \text{sort}, \text{tol}) \]
        \[
        A = Q^T A Q; R = RQ
        \]
      else
        \[
        Q = \text{Concentric}(A, i, j, \text{sort}, \text{tol}) \]
        \[
        A = Q^T A Q; R = RQ
        \]
    endif
  endfor
endfor
for \( i = n + 1 : 2n - 1 \)
  for \( j = i + 1 : 2n \)
    \[
    Q = \text{Double}(A, i, j, 0, \text{tol}) \]
    \[
    A = Q^T A Q; R = RQ
    \]
  endfor
endfor
\[
\text{rel}(A) = \frac{\text{off}(A)}{\|A\|_F}
\]
endwhile

\textbf{Algorithm 9 (Strategy S_6, S_6h).} Given a symplectic symmetric matrix \( A \in \mathbb{R}^{2n \times 2n} \), a Boolean parameter "sort", \( \text{tol} > 0 \), and \( \text{rel}(\text{tol}) > 0 \), this algorithm overwrites \( A \) with a diagonal matrix \( \Sigma = R^T A R \) where \( R \) is real symplectic orthogonal and \( \text{rel}(R^T A R) \leq \text{rel}(\text{tol}) \). If "sort" is zero, then the small angle is used at every iteration. On the other hand, if "sort" is nonzero, then the sorting angle is used when the pivot element is in the first \( n \) rows, that is, row \( i = 1, 2, \ldots, n \), and results in the first \( n \) diagonal entries of \( \Sigma \) appearing in decreasing order.
\[ A = Q^T AQ; \quad R = RQ \]
endfor
endfor

for \( i = 1 : n \)
\[
\begin{align*}
Q &= \text{Rotation}(A, i, \text{sort}, \text{tol}) \\
A &= Q^T AQ; \quad R = RQ
\end{align*}
\]
endfor

for \( i = 1 : n - 1 \)
for \( j = n + i + 1 : 2n \)
\[
\begin{align*}
Q &= \text{Concentric}(A, i, j, \text{sort}, \text{tol}) \\
A &= Q^T AQ; \quad R = RQ
\end{align*}
\]
endfor
endfor

for \( j = n + 1 : 2n - 1 \)
for \( i = j - n + 1 : n \)
\[
\begin{align*}
Q &= \text{Concentric}(A, i, j, \text{sort}, \text{tol}) \\
A &= Q^T AQ; \quad R = RQ
\end{align*}
\]
endfor
endfor

for \( i = n + 1 : 2n - 1 \)
for \( j = i + 1 : 2n \)
\[
\begin{align*}
Q &= \text{Double}(A, i, j, 0, \text{tol}) \\
A &= Q^T AQ; \quad R = RQ
\end{align*}
\]
endfor
endfor

reloff(\( A \)) = \frac{\text{off}(A)}{\|A\|_F}
endwhile

4.4 Numerical Experiments

We illustrate the experiments in two subsections, namely, structured spectral decomposition and structured SVD. All proposed algorithms were implemented in MATLAB. For matrices of size 100 × 100 or smaller, all experiments were performed on a Toshiba Satellite A45-S250 using MATLAB Version 6.5.0 (R13), while matrices of size larger than 100 × 100 were tested using MATLAB Version 7.0.0 (R14) on a Sun Ultra-10. The arithmetic was IEEE standard double precision, with a machine precision \( \text{eps} = 2.2204 \times 10^{-16} \).

Test matrices for all experiments in this section were randomly generated real symplectic symmetric matrices or real symplectic matrices using an algorithm described by Jagger in [21].
4.4.1 Structured Spectral Decomposition

The structure preserving algorithms given in Section 4.3 for computing the spectral decomposition of real symplectic symmetric matrices were performed on random real symplectic symmetric matrices. Note that, in practice, such matrices cannot be exactly real symplectic or exactly symmetric. However, the symplecticity and symmetry of our test matrices, measured via $\|AA^* - I\|_F$ and $\|A - A^T\|_F$ respectively, were of order $10^{-14}$. Since all transformations used by our algorithms are determined by using only elements in the upper triangular part of $A$, the problem of having a nearly symmetric matrix $A$ as input to our algorithms does not cause us any trouble.

In the first set of experiments, there were six algorithms tested: strategy $S_1$ (Algorithm 4), strategy $S_2$ (Algorithm 5), strategy $S_3$ (Algorithm 6), strategy $S_4$ (Algorithm 7), strategy $S_5$ (Algorithm 8), and strategy $S_6$ (Algorithm 9). All used a $\text{reloff}.\text{tol} = 10^{-13}\|A\|_F$ where $A$ is the initial matrix; the tolerance threshold for performing the Jacobi-like action, called $\text{tol}$, was chosen as $\text{tol} = 10^{-14}$. The small angle for a plane rotation was applied for these experiments. We tested all six strategies on 30 test matrices of size $30 \times 30$ with condition number 10 to decide which strategies were worthwhile for further study. Strategy $S_1$ performed extremely poorly, taking on average about 540 sweeps to bring reloff($A$) below reloff.tol. Somewhat surprisingly though, even after all these iterations, the computed eigenvalues were still in agreement with the eigenvalues computed by MATLAB: the relative error between the computed eigenvalues by strategy $S_1$ and by MATLAB was of order $10^{-13}$.

The table below gives the average number of sweeps and iterations needed by each of the other five strategies to drop reloff($A$) below reloff.tol.

<table>
<thead>
<tr>
<th>strategy</th>
<th>$S_2$</th>
<th>$S_3$</th>
<th>$S_4$</th>
<th>$S_5$</th>
<th>$S_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>sweeps</td>
<td>93.77</td>
<td>23.50</td>
<td>7.27</td>
<td>5.33</td>
<td>4.63</td>
</tr>
<tr>
<td>iterations</td>
<td>$2.22 \times 10^4$</td>
<td>$5.72 \times 10^3$</td>
<td>$1.48 \times 10^3$</td>
<td>$1.99 \times 10^3$</td>
<td>$1.71 \times 10^3$</td>
</tr>
</tbody>
</table>

Table 4.1: Average number of sweeps needed for $30 \times 30$ matrices

From Table 4.1, it can be seen that the average number of sweeps for strategies $S_2, S_3$ is too high to be considered for further study. As was mentioned in Section 4.2.6, strategy $S_6$ is obtained by reordering the sequence of pivot elements of strategy $S_5$, and from this table we see that the average number of sweeps of strategy $S_5$ and that of strategy $S_6$ are close, thus it is reasonably to choose strategy $S_5$ and strategy $S_6$ for our further work. Even though the average number of sweeps for strategy $S_4$ is higher than strategies $S_5$ and $S_6$, it is worthwhile to consider this strategy further because the number of iterations for this strategy is less than that of strategies $S_5$ and $S_6$. In fact, as was mentioned in Section...
The number of iterations per sweep for strategy \( S_4 \) is \( n^2 \) while the number of iterations per sweep for strategies \( S_5 \) and \( S_6 \) is \( 2n^2 - n \). Thus we now focus on strategies \( S_4, S_5, \) and \( S_6 \).

The second set of experiments tested variations on strategies \( S_4, S_5, \) and \( S_6 \): using the small angle for every rotation, using the sorting angle for the rotations, reducing a test matrix to bird form before applying an algorithm using the small angle rotation, and finally reducing a test matrix to bird form before applying an algorithm with the sorting angle rotation. Thus we tested twelve strategies in all. We refer to these strategies as \( S_4, S_{4s}, S_{4r}, S_{4rs}, S_5, S_{5s}, S_{5r}, S_{5rs}, S_6, S_{6s}, S_{6r}, \) and \( S_{6rs} \) where the subscript \( r \) refers to the reduction of a test matrix to bird form before applying the strategy and \( s \) denotes the use of the sorting angle. We investigated the following:

- The relationship between the condition number of test matrices and the value of \( \text{reloff.tol} \) for which each strategy drops \( \text{reloff}(A) \) below \( \text{reloff.tol} \). For this purpose, all strategies were tested on 50 test matrices of size 50 × 50 with each of the specified condition numbers: \( 10^k \) for \( k = 1, 2, \ldots, 10 \), and using \( \text{tol} = 10^{-14} \). Table 4.2 reports the result by showing the condition numbers against exponent \( m \) in the expression of \( \text{reloff.tol} = 10^{-14} \times 10^m \times \|A\|_F \) for all strategies. As can be seen, the exponent \( m \) increases when the condition number is higher than \( 10^7 \). These tests help us choose \( \text{reloff.tol} \) as a function of the condition number of the test matrix and strategy used.

<table>
<thead>
<tr>
<th>( \text{cond}(A) )</th>
<th>( S_4 )</th>
<th>( S_{4s} )</th>
<th>( S_{4r} )</th>
<th>( S_{4rs} )</th>
<th>( S_5 )</th>
<th>( S_{5s} )</th>
<th>( S_{5r} )</th>
<th>( S_{5rs} )</th>
<th>( S_6 )</th>
<th>( S_{6s} )</th>
<th>( S_{6r} )</th>
<th>( S_{6rs} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( 10^2 )</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( 10^3 )</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( 10^4 )</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( 10^5 )</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( 10^6 )</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( 10^7 )</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( 10^8 )</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>( 10^9 )</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>( 10^{10} )</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 4.2: \( \text{reloff.tol} = 10^{-14} \times 10^m \times \|A\|_F \) for the specified condition numbers

- The relationship between the condition number of test matrices and the number of sweeps needed for convergence. In this case all strategies were tested on 50 matrices of size 50 × 50 using \( \text{reloff.tol} = 10^{-13} \times \|A\|_F \), and

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
tol = 10^{-14}. The condition numbers chosen were 10, 20, 50, 100, 500, 10^3, 10^4, and 10^5. Tables 4.3 - 4.4 report the average number of sweeps of each strategy for all condition numbers under consideration. From these tables, we see that the average number of sweeps needed by each strategy increases as the condition number of the test matrix increases. Once again we also see that the average number of sweeps for the strategies using the sweep design \( S_4 \) is higher than that of strategies using the designs \( S_5 \) and \( S_6 \). As remarked earlier, the number of iterations per full sweep for the design \( S_4 \) is roughly half of that for the designs \( S_5 \) and \( S_6 \).

<table>
<thead>
<tr>
<th>( n = 50 )</th>
<th>( \text{cond}(A) )</th>
<th>( S_4 )</th>
<th>( S_{4s} )</th>
<th>( S_{4r} )</th>
<th>( S_{4rs} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>8.08</td>
<td>6.68</td>
<td>7.32</td>
<td>6.60</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>8.28</td>
<td>6.80</td>
<td>7.60</td>
<td>6.64</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>8.72</td>
<td>6.96</td>
<td>7.76</td>
<td>6.88</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>9.04</td>
<td>7.04</td>
<td>7.84</td>
<td>6.96</td>
<td></td>
</tr>
<tr>
<td>500</td>
<td>9.64</td>
<td>7.08</td>
<td>8.04</td>
<td>7.00</td>
<td></td>
</tr>
<tr>
<td>10^3</td>
<td>9.84</td>
<td>7.12</td>
<td>8.08</td>
<td>7.02</td>
<td></td>
</tr>
<tr>
<td>10^4</td>
<td>10.76</td>
<td>7.14</td>
<td>9.08</td>
<td>7.04</td>
<td></td>
</tr>
<tr>
<td>10^5</td>
<td>10.92</td>
<td>7.16</td>
<td>9.60</td>
<td>7.16</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.3: Average number of sweeps vs. condition numbers: \( S_4 \)

<table>
<thead>
<tr>
<th>( n = 50 )</th>
<th>( \text{cond}(A) )</th>
<th>( S_5 )</th>
<th>( S_{5s} )</th>
<th>( S_{5r} )</th>
<th>( S_{5rs} )</th>
<th>( S_6 )</th>
<th>( S_{6s} )</th>
<th>( S_{6r} )</th>
<th>( S_{6rs} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>5.84</td>
<td>4.10</td>
<td>5.18</td>
<td>4.08</td>
<td>4.98</td>
<td>4.44</td>
<td>4.56</td>
<td>4.06</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>6.14</td>
<td>4.36</td>
<td>5.22</td>
<td>4.26</td>
<td>5.02</td>
<td>4.58</td>
<td>4.74</td>
<td>4.10</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>6.32</td>
<td>4.42</td>
<td>5.24</td>
<td>4.40</td>
<td>5.08</td>
<td>4.76</td>
<td>4.76</td>
<td>4.14</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>6.68</td>
<td>4.66</td>
<td>5.34</td>
<td>4.50</td>
<td>5.30</td>
<td>4.78</td>
<td>4.88</td>
<td>4.24</td>
<td></td>
</tr>
<tr>
<td>500</td>
<td>7.36</td>
<td>4.86</td>
<td>5.50</td>
<td>4.86</td>
<td>5.92</td>
<td>5.06</td>
<td>4.98</td>
<td>4.48</td>
<td></td>
</tr>
<tr>
<td>10^3</td>
<td>7.90</td>
<td>5.00</td>
<td>5.68</td>
<td>4.74</td>
<td>6.08</td>
<td>5.10</td>
<td>5.06</td>
<td>4.52</td>
<td></td>
</tr>
<tr>
<td>10^4</td>
<td>8.72</td>
<td>5.26</td>
<td>5.90</td>
<td>5.00</td>
<td>6.88</td>
<td>5.20</td>
<td>5.50</td>
<td>4.92</td>
<td></td>
</tr>
<tr>
<td>10^5</td>
<td>9.80</td>
<td>5.74</td>
<td>7.10</td>
<td>5.38</td>
<td>7.54</td>
<td>5.80</td>
<td>5.74</td>
<td>5.04</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.4: Average number of sweeps vs. condition numbers: \( S_5, S_6 \)

Table 4.3 shows that \( S_{4rs} \) has the lowest average number of sweeps and \( S_4 \) has the highest, for all condition numbers under consideration. From Table 4.4, we have that \( S_{6rs} \) has the lowest average number of sweeps whereas \( S_6 \) has the highest. Plotting the common logarithm of the condition numbers against
the average number of sweeps for each strategy as shown in Figures 4.8 - 4.9, we can conclude that the number of sweeps needed by each strategy roughly increases linearly with \( \log(\text{cond}(A)) \). Observe that the graph corresponding to the strategies using the small angle has higher slope than that using the sorting angle. Tables 4.3 - 4.4 give additional information. Looking at pairs of strategies: \((S_4, S_{4s}), (S_{4r}, S_{4rs}), (S_5, S_{5s}), (S_{5r}, S_{5rs}), (S_6, S_{6s}),\) and \((S_{6r}, S_{6rs})\) where the first strategy of each pair corresponds to the strategy using the small angle for a plane rotation, and the second strategy of each pair is one with the sorting angle, we see that the average number of sweeps needed by strategies using the sorting angle is lower than that with the small angle for all pairs and for all condition numbers. We therefore focus the rest of our experiments only on strategies with sorting angle, that is, strategies \(S_{4s}, S_{4rs}, S_{5s}, S_{5rs}, S_{6s}, \) and \(S_{6rs}\).

In the third set of experiments, there were six strategies tested: \(S_{4s}, S_{4rs}, S_{5s}, S_{5rs}, S_{6s}, \) and \(S_{6rs}.\) The experiments were performed first to see additional quantities of numerical interest, and then to compare the eigenvalues computed by these strategies against those computed by MATLAB.

To achieve the first goal, 50 test matrices of dimension \(2n \times 2n\) for \(n = 25, 50, 75, 100\) with condition number 10 were generated. All strategies were tested on each of these matrices using \(\text{reloff.tol} = 10^{-13} \times \|A\|_F\) and \(\text{tol} = 10^{-14}\). Some of results we obtained are reported in the following.

- **Sweeps** denotes the average number of sweeps needed by each strategy to drop \(\text{reloff}(A)\) below \(\text{reloff.tol}\). Table 4.5 provides the low, high, and average number of sweeps needed by all strategies and all matrix sizes under consideration. From this table, we see that the average number of sweeps increases with increasing matrix sizes for all strategies. We notice that, the standard deviation of the number of sweeps was consistently very low — between 0 and 0.5.

<table>
<thead>
<tr>
<th>2n</th>
<th>50</th>
<th>100</th>
<th>150</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strategy</td>
<td>low</td>
<td>high</td>
<td>avg.</td>
<td>low</td>
</tr>
<tr>
<td>(S_{4s})</td>
<td>7</td>
<td>9</td>
<td>7.22</td>
<td>7</td>
</tr>
<tr>
<td>(S_{4rs})</td>
<td>6</td>
<td>9</td>
<td>7.24</td>
<td>7</td>
</tr>
<tr>
<td>(S_{5s})</td>
<td>4</td>
<td>5</td>
<td>4.28</td>
<td>4</td>
</tr>
<tr>
<td>(S_{5rs})</td>
<td>4</td>
<td>5</td>
<td>4.24</td>
<td>4</td>
</tr>
<tr>
<td>(S_{6s})</td>
<td>4</td>
<td>5</td>
<td>4.46</td>
<td>5</td>
</tr>
<tr>
<td>(S_{6rs})</td>
<td>4</td>
<td>5</td>
<td>4.06</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 4.5: Statistical data for the number of sweeps
Figure 4.8: Average number of sweeps vs. condition numbers: \( S_4 \). Note that the number of iterations in a full sweep of \( S_4 \) is \( n^2 \).

Figure 4.9: Average number of sweeps vs. condition numbers: \( S_5, S_6 \). Note that the number of iterations in a full sweep of \( S_5 \) or \( S_6 \) is \( 2n^2 - n \).
Even though the average number of sweeps for strategies $S_4s$ and $S_4rs$ is higher than the other four strategies for all matrix sizes under consideration, the total number of iterations needed by strategies $S_4s$ and $S_4rs$ as shown in Table 4.6 is lower than the other four strategies. Here we present only the low, high, and average total number of iterations for test matrices of dimension $150 \times 150$ and $200 \times 200$. Moreover, from Table 4.6, if we compare the average total number of iterations for test matrices of dimension $200 \times 200$ among strategies $S_4s, S_5s,$ and $S_6s$, then we have that the average total number of iterations of $S_4s$ is about $15.72\%$ and $14.78\%$ less than that of $S_5s$ and $S_6s$, respectively. Comparing the average total number of iterations for test matrices of dimension $200 \times 200$ among strategies $S_4rs, S_5rs,$ and $S_6rs$, we have the average total number of iterations of $S_4rs$ is about $27.70\%$ and $19.49\%$ less than that of $S_5rs$ and $S_6rs$, respectively.

<table>
<thead>
<tr>
<th>2n</th>
<th>Strategy</th>
<th>low</th>
<th>high</th>
<th>avg.</th>
<th>low</th>
<th>high</th>
<th>avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>150</td>
<td>$S_4s$</td>
<td>$3.70 \times 10^4$</td>
<td>$3.84 \times 10^4$</td>
<td>$3.74 \times 10^4$</td>
<td>$6.74 \times 10^4$</td>
<td>$6.99 \times 10^4$</td>
<td>$6.86 \times 10^4$</td>
</tr>
<tr>
<td></td>
<td>$S_4rs$</td>
<td>$3.74 \times 10^4$</td>
<td>$4.03 \times 10^4$</td>
<td>$3.86 \times 10^4$</td>
<td>$7.13 \times 10^4$</td>
<td>$7.45 \times 10^4$</td>
<td>$7.23 \times 10^4$</td>
</tr>
<tr>
<td></td>
<td>$S_5s$</td>
<td>$4.49 \times 10^4$</td>
<td>$4.64 \times 10^4$</td>
<td>$4.55 \times 10^4$</td>
<td>$8.03 \times 10^4$</td>
<td>$8.27 \times 10^4$</td>
<td>$8.14 \times 10^4$</td>
</tr>
<tr>
<td></td>
<td>$S_5rs$</td>
<td>$4.22 \times 10^4$</td>
<td>$4.31 \times 10^4$</td>
<td>$4.28 \times 10^4$</td>
<td>$9.45 \times 10^4$</td>
<td>$1.09 \times 10^5$</td>
<td>$1.00 \times 10^5$</td>
</tr>
<tr>
<td></td>
<td>$S_6s$</td>
<td>$4.80 \times 10^4$</td>
<td>$5.01 \times 10^4$</td>
<td>$4.89 \times 10^4$</td>
<td>$9.46 \times 10^4$</td>
<td>$9.06 \times 10^4$</td>
<td>$8.89 \times 10^4$</td>
</tr>
<tr>
<td></td>
<td>$S_6rs$</td>
<td>$4.53 \times 10^4$</td>
<td>$4.85 \times 10^4$</td>
<td>$4.67 \times 10^4$</td>
<td>$7.86 \times 10^4$</td>
<td>$8.44 \times 10^4$</td>
<td>$8.05 \times 10^4$</td>
</tr>
</tbody>
</table>

Table 4.6: Statistical data for the total number of iterations

- To compare the cost required by each strategy, we calculate the cost of updating $A$ per full sweep. As discussed earlier, if a real symplectic plane rotation is used for a similarity, then the cost of updating $A$ is $O(4n)$, while the update cost when using a real symplectic double rotation for a similarity is $O(8n)$. Table 4.7 shows the total cost per full sweep required by strategies $S_4, S_5,$ and $S_6$. As we see, a full sweep $S_4$ incurs approximately half of the cost of a full sweep of $S_5$ or $S_6$.

- For all strategies, reloff($A$) in practice converges quadratically. The typical behavior of reloff($A$) for a random real $200 \times 200$ symplectic symmetric matrix is shown in Figure 4.10.

- As the generated matrix $A$ is real symplectic symmetric, its symplecticity and symmetry were tested via $\|AA^* - I\|_F$ and $\|A - A^T\|_F$, respectively. It can be seen from Tables 4.8 - 4.13 that all strategies were run on well structured matrices. However, there was some loss of symplecticity of the
Figure 4.10: Typical convergence behavior of a random $200 \times 200$ real symplectic symmetric matrix. Note that a full sweep of $S_{4s}$ and $S_{4rs}$ has roughly half as many iterations as a full sweep of other strategies.
strategy & $S_4$ & $S_5$ & $S_6$ \\
number of iterations per sweep & $n^2$ & $2n^2 - n$ & $2n^2 - n$ \\
plane rotations per sweep & $n$ & $n$ & $n$ \\
double rotations per sweep & $\frac{n(n-1)}{2}$ & $n^2 - n$ & $n^2 - n$ \\
concentric rotations per sweep & $\frac{n(n-1)}{2}$ & $n^2 - n$ & $n^2 - n$ \\
total cost per sweep & $O(8n^3 - 4n^2)$ & $O(16n^3 - 12n^2)$ & $O(16n^3 - 12n^2)$ \\

Table 4.7: The maximum cost per sweep for $S_4$, $S_5$ and $S_6$

generated test matrices when the dimension increases. In particular, the loss of symplecticity of the test matrices of dimension $50 \times 50$ was of order $10^{-14}$ while for matrices of dimension $200 \times 200$ this loss is of order $10^{-13}$.

- Each strategy computes $R$ which is an approximation to a real symplectic orthogonal matrix of eigenvectors of the test matrix. In order to see how well the computed $R$ has both structures, $\|RR^T - I\|_F$ and $\|RR^T - I\|_F$ were calculated. As can be seen from Tables 4.8 - 4.13, these quantities were, on average, of the order $10^{-13}$. So our algorithms provided well structured of $R$.

- The deviation of $\Sigma$ from symplectic structure is measured by $\text{diag.alg} = \max |\text{diag}(S_1) \ast \text{diag}(S_2) - 1|$ where $S_1 = \Sigma(1 : n, 1 : n)$, $S_2 = \Sigma(n + 1 : 2n, n + 1 : 2n)$, and $\ast$ denotes array multiplication (as in MATLAB). It can be seen from Tables 4.8 - 4.13 that $\Sigma$ computed by our strategies is symplectic to within $1.08 \times 10^{-13}$.

- The row labeled $\text{rel.eig}$ records the maximum relative error as measured by

$$\text{releig} = \max_j \left| \frac{\lambda_j^{\text{alg}} - \lambda_j^{\text{eig}}}{\lambda_j^{\text{eig}}} \right|,$$  \hspace{1cm} (4.26)

where $\{\lambda_j^{\text{alg}}\}$ and $\{\lambda_j^{\text{eig}}\}$ denote the eigenvalues computed by MATLAB's $\text{eig}$ function and by our strategies, respectively. As can be seen from Tables 4.8 - 4.13, the eigenvalues computed by our strategies when compared to MATLAB had relative error around $10^{-14}$, for all dimensions under consideration.

The next set of experiments were designed to see how accurately our strategies computed the eigenvalues. To investigate this, each strategy was tested on a real symplectic symmetric test matrix with pre-chosen eigenvalues. Then we compared the known eigenvalues to eigenvalues computed by our strategies and to those computed by MATLAB's $\text{eig}$ function. Some of the experiments are shown below.
<table>
<thead>
<tr>
<th>2n</th>
<th>50</th>
<th>100</th>
<th>150</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td>sweeps</td>
<td>7.22</td>
<td>7.96</td>
<td>8.38</td>
<td>8.75</td>
</tr>
<tr>
<td>reloff</td>
<td>$4.90 \times 10^{-14}$</td>
<td>$6.33 \times 10^{-14}$</td>
<td>$2.01 \times 10^{-13}$</td>
<td>$1.20 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|AA^* - I|_F$</td>
<td>$1.90 \times 10^{-14}$</td>
<td>$4.64 \times 10^{-14}$</td>
<td>$8.17 \times 10^{-14}$</td>
<td>$1.23 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|A - A^T|_F$</td>
<td>$1.83 \times 10^{-15}$</td>
<td>$3.19 \times 10^{-15}$</td>
<td>$4.34 \times 10^{-15}$</td>
<td>$5.32 \times 10^{-15}$</td>
</tr>
<tr>
<td>$|RR^* - I|_F$</td>
<td>$5.81 \times 10^{-14}$</td>
<td>$1.72 \times 10^{-13}$</td>
<td>$3.29 \times 10^{-13}$</td>
<td>$5.22 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|RR^T - I|_F$</td>
<td>$5.81 \times 10^{-14}$</td>
<td>$1.72 \times 10^{-13}$</td>
<td>$3.29 \times 10^{-13}$</td>
<td>$5.22 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$2.31 \times 10^{-14}$</td>
<td>$4.64 \times 10^{-14}$</td>
<td>$7.04 \times 10^{-14}$</td>
<td>$9.34 \times 10^{-14}$</td>
</tr>
<tr>
<td>releig</td>
<td>$1.38 \times 10^{-14}$</td>
<td>$2.54 \times 10^{-14}$</td>
<td>$3.84 \times 10^{-14}$</td>
<td>$5.04 \times 10^{-14}$</td>
</tr>
</tbody>
</table>

Table 4.8: Strategy $S_{4s}$

<table>
<thead>
<tr>
<th>2n</th>
<th>50</th>
<th>100</th>
<th>150</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td>sweeps</td>
<td>7.24</td>
<td>8.18</td>
<td>8.82</td>
<td>9.10</td>
</tr>
<tr>
<td>reloff</td>
<td>$2.78 \times 10^{-14}$</td>
<td>$3.05 \times 10^{-14}$</td>
<td>$1.59 \times 10^{-13}$</td>
<td>$7.87 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|AA^* - I|_F$</td>
<td>$1.90 \times 10^{-14}$</td>
<td>$4.64 \times 10^{-14}$</td>
<td>$8.17 \times 10^{-14}$</td>
<td>$1.23 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|A - A^T|_F$</td>
<td>$1.83 \times 10^{-15}$</td>
<td>$3.19 \times 10^{-15}$</td>
<td>$4.34 \times 10^{-15}$</td>
<td>$5.32 \times 10^{-15}$</td>
</tr>
<tr>
<td>$|RR^* - I|_F$</td>
<td>$5.96 \times 10^{-14}$</td>
<td>$1.93 \times 10^{-13}$</td>
<td>$3.69 \times 10^{-13}$</td>
<td>$5.93 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|RR^T - I|_F$</td>
<td>$5.96 \times 10^{-14}$</td>
<td>$1.93 \times 10^{-13}$</td>
<td>$3.69 \times 10^{-13}$</td>
<td>$5.93 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$2.43 \times 10^{-14}$</td>
<td>$5.20 \times 10^{-14}$</td>
<td>$7.85 \times 10^{-14}$</td>
<td>$1.06 \times 10^{-13}$</td>
</tr>
<tr>
<td>releig</td>
<td>$1.47 \times 10^{-14}$</td>
<td>$2.88 \times 10^{-14}$</td>
<td>$4.27 \times 10^{-14}$</td>
<td>$5.62 \times 10^{-14}$</td>
</tr>
</tbody>
</table>

Table 4.9: Strategy $S_{4rs}$

<table>
<thead>
<tr>
<th>2n</th>
<th>50</th>
<th>100</th>
<th>150</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td>sweeps</td>
<td>4.28</td>
<td>4.90</td>
<td>5.00</td>
<td>5.16</td>
</tr>
<tr>
<td>reloff</td>
<td>$5.54 \times 10^{-14}$</td>
<td>$9.27 \times 10^{-14}$</td>
<td>$1.56 \times 10^{-13}$</td>
<td>$1.19 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|AA^* - I|_F$</td>
<td>$1.94 \times 10^{-14}$</td>
<td>$4.55 \times 10^{-14}$</td>
<td>$8.16 \times 10^{-14}$</td>
<td>$1.24 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|A - A^T|_F$</td>
<td>$1.73 \times 10^{-15}$</td>
<td>$3.28 \times 10^{-15}$</td>
<td>$4.24 \times 10^{-15}$</td>
<td>$5.36 \times 10^{-15}$</td>
</tr>
<tr>
<td>$|RR^* - I|_F$</td>
<td>$6.53 \times 10^{-14}$</td>
<td>$2.23 \times 10^{-13}$</td>
<td>$3.87 \times 10^{-13}$</td>
<td>$6.20 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|RR^T - I|_F$</td>
<td>$6.53 \times 10^{-14}$</td>
<td>$2.23 \times 10^{-13}$</td>
<td>$3.87 \times 10^{-13}$</td>
<td>$6.20 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$2.64 \times 10^{-14}$</td>
<td>$6.19 \times 10^{-14}$</td>
<td>$7.85 \times 10^{-14}$</td>
<td>$1.08 \times 10^{-13}$</td>
</tr>
<tr>
<td>releig</td>
<td>$1.51 \times 10^{-14}$</td>
<td>$3.36 \times 10^{-14}$</td>
<td>$4.27 \times 10^{-14}$</td>
<td>$5.80 \times 10^{-14}$</td>
</tr>
</tbody>
</table>

Table 4.10: Strategy $S_{5s}$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
<table>
<thead>
<tr>
<th>2n</th>
<th>50</th>
<th>100</th>
<th>150</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td>sweeps</td>
<td>4.24</td>
<td>4.90</td>
<td>5.04</td>
<td>5.48</td>
</tr>
<tr>
<td>reloff</td>
<td>$3.65 \times 10^{-14}$</td>
<td>$9.27 \times 10^{-14}$</td>
<td>$2.26 \times 10^{-14}$</td>
<td>$1.41 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|AA^*-I|_F$</td>
<td>$1.94 \times 10^{-14}$</td>
<td>$4.55 \times 10^{-14}$</td>
<td>$8.16 \times 10^{-14}$</td>
<td>$1.24 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|A-A^T|_F$</td>
<td>$1.73 \times 10^{-15}$</td>
<td>$3.28 \times 10^{-15}$</td>
<td>$4.24 \times 10^{-15}$</td>
<td>$5.36 \times 10^{-15}$</td>
</tr>
<tr>
<td>$|RR^*-I|_F$</td>
<td>$6.86 \times 10^{-14}$</td>
<td>$2.23 \times 10^{-13}$</td>
<td>$4.31 \times 10^{-13}$</td>
<td>$7.00 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|RR^T-I|_F$</td>
<td>$6.86 \times 10^{-14}$</td>
<td>$2.23 \times 10^{-13}$</td>
<td>$4.31 \times 10^{-13}$</td>
<td>$7.00 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$2.77 \times 10^{-14}$</td>
<td>$6.19 \times 10^{-14}$</td>
<td>$9.27 \times 10^{-14}$</td>
<td>$1.29 \times 10^{-13}$</td>
</tr>
<tr>
<td>releig</td>
<td>$1.59 \times 10^{-14}$</td>
<td>$3.36 \times 10^{-14}$</td>
<td>$4.91 \times 10^{-14}$</td>
<td>$6.80 \times 10^{-14}$</td>
</tr>
</tbody>
</table>

Table 4.11: Strategy $S_{5rs}$

<table>
<thead>
<tr>
<th>2n</th>
<th>50</th>
<th>100</th>
<th>150</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td>sweeps</td>
<td>4.46</td>
<td>5.02</td>
<td>5.08</td>
<td>5.54</td>
</tr>
<tr>
<td>reloff</td>
<td>$4.36 \times 10^{-14}$</td>
<td>$1.18 \times 10^{-14}$</td>
<td>$2.23 \times 10^{-14}$</td>
<td>$1.21 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|AA^*-I|_F$</td>
<td>$1.94 \times 10^{-14}$</td>
<td>$4.55 \times 10^{-14}$</td>
<td>$8.16 \times 10^{-14}$</td>
<td>$1.24 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|A-A^T|_F$</td>
<td>$1.73 \times 10^{-15}$</td>
<td>$3.28 \times 10^{-15}$</td>
<td>$4.24 \times 10^{-15}$</td>
<td>$5.36 \times 10^{-15}$</td>
</tr>
<tr>
<td>$|RR^*-I|_F$</td>
<td>$6.61 \times 10^{-14}$</td>
<td>$2.103 \times 10^{-13}$</td>
<td>$4.29 \times 10^{-13}$</td>
<td>$6.87 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|RR^T-I|_F$</td>
<td>$6.61 \times 10^{-14}$</td>
<td>$2.103 \times 10^{-13}$</td>
<td>$4.29 \times 10^{-13}$</td>
<td>$6.87 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$2.60 \times 10^{-14}$</td>
<td>$5.44 \times 10^{-14}$</td>
<td>$8.80 \times 10^{-14}$</td>
<td>$1.21 \times 10^{-13}$</td>
</tr>
<tr>
<td>releig</td>
<td>$1.51 \times 10^{-14}$</td>
<td>$2.99 \times 10^{-14}$</td>
<td>$4.71 \times 10^{-14}$</td>
<td>$6.41 \times 10^{-14}$</td>
</tr>
</tbody>
</table>

Table 4.12: Strategy $S_{6s}$

<table>
<thead>
<tr>
<th>2n</th>
<th>50</th>
<th>100</th>
<th>150</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td>sweeps</td>
<td>4.06</td>
<td>4.80</td>
<td>5.04</td>
<td>5.20</td>
</tr>
<tr>
<td>reloff</td>
<td>$2.41 \times 10^{-14}$</td>
<td>$4.44 \times 10^{-15}$</td>
<td>$1.63 \times 10^{-14}$</td>
<td>$1.38 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|AA^*-I|_F$</td>
<td>$1.94 \times 10^{-14}$</td>
<td>$4.55 \times 10^{-14}$</td>
<td>$8.16 \times 10^{-14}$</td>
<td>$1.24 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|A-A^T|_F$</td>
<td>$1.73 \times 10^{-15}$</td>
<td>$3.28 \times 10^{-15}$</td>
<td>$4.24 \times 10^{-15}$</td>
<td>$5.36 \times 10^{-15}$</td>
</tr>
<tr>
<td>$|RR^*-I|_F$</td>
<td>$6.51 \times 10^{-14}$</td>
<td>$2.24 \times 10^{-13}$</td>
<td>$4.63 \times 10^{-13}$</td>
<td>$7.56 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|RR^T-I|_F$</td>
<td>$6.51 \times 10^{-14}$</td>
<td>$2.24 \times 10^{-13}$</td>
<td>$4.63 \times 10^{-13}$</td>
<td>$7.56 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$2.70 \times 10^{-14}$</td>
<td>$6.11 \times 10^{-14}$</td>
<td>$9.80 \times 10^{-14}$</td>
<td>$1.39 \times 10^{-13}$</td>
</tr>
<tr>
<td>releig</td>
<td>$1.54 \times 10^{-14}$</td>
<td>$3.31 \times 10^{-14}$</td>
<td>$5.25 \times 10^{-14}$</td>
<td>$7.28 \times 10^{-14}$</td>
</tr>
</tbody>
</table>

Table 4.13: Strategy $S_{6rs}$

81

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Example 1. A real symplectic symmetric matrix $A$ was generated with eigenvalues $10, 5, 2, \frac{1}{2}, \frac{1}{5}, \frac{1}{10}$. In this case the given matrix $A$ is a well-conditioned matrix having distinct eigenvalues. Table 4.14 shows the exact eigenvalues and the relative errors in the eigenvalues approximations computed by our algorithms and by MATLAB’s $eig$ function.

![Table 4.14](image)

Table 4.14: Accuracy of the eigenvalues $10, 5, 2, \frac{1}{2}, \frac{1}{5}, \frac{1}{10}$

Example 2. A real symplectic symmetric matrix $A$ was generated with eigenvalue $10, 5, 5, \frac{1}{5}, \frac{1}{5}, \frac{1}{10}$. Here $A$ is a well-conditioned matrix having two set of repeated eigenvalues. The exact eigenvalues and the relative errors are reported in Table 4.15.

![Table 4.15](image)

Table 4.15: Accuracy of the eigenvalues $10, 5, 5, \frac{1}{5}, \frac{1}{5}, \frac{1}{10}$

Example 3. A real symplectic symmetric matrix $A$ was generated with eigenvalues $a, b, c, \frac{1}{a}, \frac{1}{b}, \frac{1}{c}$ where $[a, b, c] = [3, 3, 3] + [6.2, 5.4, 3.1] \times 10^{-15}$. Here $A$ is a well-conditioned matrix and it has clusters of eigenvalues near $3$ and $\frac{1}{3}$. The exact eigenvalues and the relative errors are reported in Table 4.16.

Example 4. A real symplectic symmetric matrix $A$ was generated with eigenvalues $10^4, 10^2, 10, \frac{1}{10}, \frac{1}{10^2}, \frac{1}{10^4}$. In this case, $A$ is an ill-conditioned matrix. The result of experiments reports in Table 4.17.

Tables 4.14 - 4.17 show that the eigenvalues computed by each strategy are comparable. In Examples 1 - 3, $A$ is a well-conditioned matrix, and all four strategies yield the exact eigenvalues with relative error close to machine precision as
Table 4.16: Accuracy of the eigenvalues near 3 and \( \frac{1}{3} \)

<table>
<thead>
<tr>
<th>( \lambda_j )</th>
<th>( S_{4s} )</th>
<th>( S_{4rs} )</th>
<th>( S_{5s} )</th>
<th>( S_{5rs} )</th>
<th>( S_{6s} )</th>
<th>( S_{6rs} )</th>
<th>Matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/a</td>
<td>6.7e-16</td>
<td>5.0e-16</td>
<td>9.9e-16</td>
<td>1.7e-16</td>
<td>1.7e-16</td>
<td>8.3e-16</td>
<td>1.3e-15</td>
</tr>
<tr>
<td>1/b</td>
<td>6.7e-16</td>
<td>5.0e-16</td>
<td>1.7e-16</td>
<td>3.3e-16</td>
<td>0</td>
<td>5.0e-16</td>
<td>1.2e-15</td>
</tr>
<tr>
<td>1/c</td>
<td>1.7e-16</td>
<td>1.7e-16</td>
<td>1.7e-16</td>
<td>5.0e-16</td>
<td>3.3e-16</td>
<td>8.3e-16</td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>3.0e-16</td>
<td>1.5e-16</td>
<td>1.5e-16</td>
<td>3.0e-16</td>
<td>3.0e-16</td>
<td>5.9e-16</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>7.4e-16</td>
<td>4.4e-16</td>
<td>7.4e-16</td>
<td>1.5e-16</td>
<td>0</td>
<td>3.0e-16</td>
<td>4.4e-16</td>
</tr>
<tr>
<td>a</td>
<td>8.9e-16</td>
<td>1.5e-16</td>
<td>1.5e-16</td>
<td>0</td>
<td>1.5e-16</td>
<td>3.0e-16</td>
<td>7.4e-16</td>
</tr>
</tbody>
</table>

Table 4.17: Accuracy of the eigenvalues \( 10^4, 10^2, 10, \frac{1}{10}, \frac{1}{10^2}, \frac{1}{10^4} \)

<table>
<thead>
<tr>
<th>( \lambda_j )</th>
<th>( S_{4s} )</th>
<th>( S_{4rs} )</th>
<th>( S_{5s} )</th>
<th>( S_{5rs} )</th>
<th>( S_{6s} )</th>
<th>( S_{6rs} )</th>
<th>Matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/10^4</td>
<td>3.8e-11</td>
<td>4.3e-11</td>
<td>1.0e-9</td>
<td>6.1e-11</td>
<td>1.0e-9</td>
<td>5.9e-11</td>
<td>1.6e-9</td>
</tr>
<tr>
<td>1/10^2</td>
<td>4.0e-12</td>
<td>3.0e-11</td>
<td>2.1e-12</td>
<td>2.2e-11</td>
<td>7.0e-12</td>
<td>2.0e-11</td>
<td>5.1e-11</td>
</tr>
<tr>
<td>1/10</td>
<td>3.5e-12</td>
<td>6.6e-13</td>
<td>4.0e-13</td>
<td>8.9e-13</td>
<td>9.3e-13</td>
<td>1.0e-12</td>
<td>6.6e-13</td>
</tr>
<tr>
<td>10</td>
<td>3.7e-14</td>
<td>2.3e-14</td>
<td>3.0e-14</td>
<td>3.4e-14</td>
<td>2.7e-14</td>
<td>3.7e-14</td>
<td>1.1e-13</td>
</tr>
<tr>
<td>10^2</td>
<td>6.0e-14</td>
<td>1.1e-14</td>
<td>4.8e-15</td>
<td>1.4e-15</td>
<td>1.1e-14</td>
<td>3.1e-15</td>
<td>7.1e-15</td>
</tr>
<tr>
<td>10^4</td>
<td>7.3e-16</td>
<td>9.1e-16</td>
<td>1.8e-15</td>
<td>9.1e-16</td>
<td>1.3e-15</td>
<td>1.3e-15</td>
<td>7.3e-16</td>
</tr>
</tbody>
</table>

As mentioned earlier in Sections 2.3.1 and 2.3.2, during iterations where real symplectic double rotations are used, there could possibly be an increase in the off-diagonal norm. Even though all six strategies \( S_{4s}, S_{4rs}, S_{5s}, S_{5rs}, S_{6s}, S_{6rs} \), use such tools, numerical experiments show that these strategies always drop reloff below \( \text{reloff.tol} \). We finish this section by computing the number of iterations per sweep where the off-diagonal norm increases for each strategy. Tests were run on a typical real symplectic symmetric matrix of size \( 50 \times 50 \) with condition number 10. These tests used \( \text{reloff.tol} = 10^{-13} \times \|A\|_F \) and \( \text{tol} = 10^{-14} \). The results we obtained are reported in Table 4.18, in which the following notation is used:

\[
\begin{align*}
D & := \text{Number of iterations performed during a sweep.} \\
\overline{d} & := \text{Number of iterations in a sweep during which } \text{off}(A) \text{ increases.} \\
\overline{d}_1 & := \text{Number of iterations in a sweep when a real symplectic double rotation with direct sum embedding is used and } \text{off}(A) \text{ increases.} \\
\overline{d}_2 & := \text{Number of iterations in a sweep when a real symplectic double rotation with concentric embedding is used and } \text{off}(A) \text{ increases.}
\end{align*}
\]

Observe that \( \overline{d}_1 + \overline{d}_2 = \overline{d} < D \). Table 4.18 also reports the percentages \( \frac{\overline{d}_1}{D}, \frac{\overline{d}_2}{D}, \frac{\overline{d}}{D} \).

We see that reloff decreases with each sweep. Even though, during a sweep,
<table>
<thead>
<tr>
<th>strategy</th>
<th>sweep</th>
<th>$D$</th>
<th>reloff</th>
<th># iters. where off($A$) increases</th>
<th>percentage of increase</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$d_1$, $d_2$, $d$, $\frac{d_1}{d}$%</td>
<td>$\frac{d_2}{d}$%</td>
</tr>
<tr>
<td>$S_{4s}$</td>
<td>1</td>
<td>625</td>
<td>$3.1 \times 10^{-1}$</td>
<td>14 59 73</td>
<td>2.24 9.44 11.68</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>625</td>
<td>$1.0 \times 10^{-1}$</td>
<td>8 13 21</td>
<td>1.28 2.08 3.36</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>625</td>
<td>$1.8 \times 10^{-2}$</td>
<td>0 1 1</td>
<td>0.16 0.16</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>625</td>
<td>$3.3 \times 10^{-3}$</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>625</td>
<td>$3.7 \times 10^{-4}$</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>539</td>
<td>$5.0 \times 10^{-7}$</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>96</td>
<td>$2.2 \times 10^{-12}$</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>6</td>
<td>$6.3 \times 10^{-15}$</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td>$S_{3rs}$</td>
<td>1</td>
<td>625</td>
<td>$2.1 \times 10^{-1}$</td>
<td>13 46 59</td>
<td>2.08 7.36 9.44</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>625</td>
<td>$6.8 \times 10^{-2}$</td>
<td>0 6 6</td>
<td>0 0.96 0.96</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>625</td>
<td>$1.7 \times 10^{-2}$</td>
<td>0 2 2</td>
<td>0 0.32 0.32</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>625</td>
<td>$2.0 \times 10^{-3}$</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>625</td>
<td>$5.4 \times 10^{-5}$</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>496</td>
<td>$4.5 \times 10^{-8}$</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>57</td>
<td>$6.3 \times 10^{-15}$</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td>$S_{5s}$</td>
<td>1</td>
<td>1225</td>
<td>$2.0 \times 10^{-1}$</td>
<td>91 48 139</td>
<td>7.43 3.92 11.35</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1225</td>
<td>$1.5 \times 10^{-2}$</td>
<td>39 6 45</td>
<td>3.18 0.49 3.67</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1225</td>
<td>$2.1 \times 10^{-4}$</td>
<td>20 2 22</td>
<td>1.63 0.16 1.80</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1019</td>
<td>$1.5 \times 10^{-12}$</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>47</td>
<td>$8.1 \times 10^{-15}$</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td>$S_{5rs}$</td>
<td>1</td>
<td>1225</td>
<td>$1.4 \times 10^{-1}$</td>
<td>70 19 89</td>
<td>5.71 1.55 7.27</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1225</td>
<td>$3.8 \times 10^{-3}$</td>
<td>42 5 47</td>
<td>3.43 0.41 3.84</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1225</td>
<td>$2.4 \times 10^{-5}$</td>
<td>5 2 7</td>
<td>0.41 0.16 0.57</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1059</td>
<td>$4.8 \times 10^{-15}$</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td>$S_{6s}$</td>
<td>1</td>
<td>1225</td>
<td>$1.6 \times 10^{-1}$</td>
<td>50 61 11</td>
<td>4.08 4.98 9.06</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1225</td>
<td>$2.0 \times 10^{-2}$</td>
<td>11 35 46</td>
<td>0.90 3.86 3.76</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1225</td>
<td>$4.4 \times 10^{-4}$</td>
<td>1 1 2</td>
<td>0.08 0.08 0.16</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1199</td>
<td>$1.5 \times 10^{-9}$</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>254</td>
<td>$7.0 \times 10^{-15}$</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td>$S_{6rs}$</td>
<td>1</td>
<td>1225</td>
<td>$9.1 \times 10^{-2}$</td>
<td>46 21 67</td>
<td>3.76 1.71 5.47</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1225</td>
<td>$8.8 \times 10^{-3}$</td>
<td>7 25 32</td>
<td>0.57 2.04 2.61</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1225</td>
<td>$8.7 \times 10^{-6}$</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>755</td>
<td>$2.2 \times 10^{-14}$</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
</tbody>
</table>

Table 4.18: Measuring the non-monotonicity of off($A$) when $A$ is a typical real 50 $\times$ 50 symplectic symmetric matrix.
there are some errant iterations during which the off-diagonal weight in fact increases. Observe also that these errant iterations only occur during the early sweeps. Table 4.18 also tells us that we should take a closer look at the behavior of \( \text{off}(A) \) during the first two sweeps. In Figures 4.11 - 4.13, \( y = \frac{\text{off}(A^{(i)}) - \text{off}(A^{(i-1)})}{\text{off}(A^{(i-1)})} \) is plotted against the iteration numbers, for \( S_{4s} \) using a typical real 50 \( \times \) 50 symplectic symmetric matrix with the condition number 10. Note that negative values of \( y \) correspond to the good iterations, i.e. those when \( \text{off}(A) \) decreases. The graph clearly shows that the "good" iterations dominate; even at iterations when \( \text{off}(A) \) increases, this increase is on average far smaller than the decrease of \( \text{off}(A) \) at an iteration.

4.4.2 Structured SVD

In this section we present numerical experiments for computing a real symplectic SVD of a real symplectic matrix \( A \). As mentioned earlier, the real symplectic of \( A \) can be constructed from the structured spectral decomposition as in (1.16). That is,

\[
A = U H, \quad \text{(polar decomposition)}
\]

\[
= (UQ) \Sigma Q^T \quad \text{where} \quad H = Q \Sigma Q^T,
\]

\[
= W \Sigma Q^T \quad \text{with} \quad W = UQ. \tag{4.27}
\]

All experiments here comprise two steps: a step in which the real symplectic orthogonal matrix \( U \) is computed, and a step in which the structured spectral decomposition of the real symplectic symmetric \( H \) is computed.

As remarked at the beginning of this Chapter, we used the Newton iteration as given in (4.1) to compute \( U \). Our first set of experiments in this section investigated the number of iterations needed by Newton iteration to make \( X_{k+1} \) closer to \( U \) and to measure the deviation of \( U \) from symplecticity and orthogonality. The stopping criterion used for the Newton iteration was

\[
\|X_{k+1} - X_k\| < 10^{-14}.
\]

The Newton iteration was performed on 100 random real symplectic matrices of dimension \( 2n \times 2n \) for \( n = 25, 50, 75, 100 \) with condition number 10. Experimental results show that only 7 iterations were required for any test matrix. The average symplecticity of test matrices and the average departure of being real symplectic and orthogonal of computed matrices \( U \) are reported in Table 4.19. As we see, the computed matrices \( U \) were both real symplectic and orthogonal to within \( 1.15 \times 10^{-13} \).

Now the computation of the polar factor \( U \) is completed. To achieve the real symplectic SVD of \( A \), we need to compute the structured spectral decomposition
Figure 4.11: Relative change in off during first two sweeps

Figure 4.12: Detail of first sweep

Figure 4.13: Detail of second sweep
of the real symplectic symmetric factor $H = U^T A$. Our algorithms for computing such a decomposition have been previously described in Section 4.4.1. As before, we used only four strategies: $S_{5r}, S_{5rs}, S_{6r},$ and $S_{6rs}$ for all experiments presented in this section.

Combining two steps — computing the real symplectic orthogonal $U$ and computing the real symplectic symmetric $H$ — we now have the desired real symplectic SVD.

All experiments presented here were performed to test all structured factors of the real symplectic SVD of $A$ as in (4.27) and to see additional quantities of numerical interest. Fifty real symplectic matrices of dimension $2n \times 2n$ for $n = 25, 50, 75, 100$, with condition number 10 were generated. All algorithms were run on each of these matrices using $\text{reloff.tol} = 10^{-13} \times ||A||_F$ and $\text{tol} = 10^{-14}$.

The results we obtained are shown in Tables 4.20 - 4.25 and discussed below.

- The symplecticity of the test matrix $A$ was measured via $||AA^* - I||_F$. We see that all strategies under consideration were tested on matrices that were symplectic to within $5.33 \times 10^{-14}$.

- $\text{Sweeps}$ denotes the average number of sweeps needed by each strategy to bring $\text{reloff}(A)$ below $\text{reloff.tol}$. It can be seen that for all strategies this number increases as the size of the test matrix increases.

- $\text{Reloff}$ reports the relative off-diagonal norm of the last iteration for each strategy before it terminated, showing that our strategies always drop the reloff below $\text{reloff.tol}$.

- $||UU^* - I||_F$ and $||UUT - I||_F$ measure the average deviation from symplecticity and orthogonality, respectively of the $U$ computed by Newton iteration (4.1). As we see the computed matrices $U$ were both real symplectic and orthogonal to within $1.12 \times 10^{-13}$. Also, the departure from being real symplectic and symmetric of the factor $H = U^T A$ are measured via $||HH^* - I||_F$ and $||H - HT||_F$, respectively. We see that the computed matrices $H$ were both real symplectic as well as symmetric to within order $10^{-14}$.

- To see how well our strategies compute the real symplectic orthogonal matrices $W$ and $Q$ of (4.27), we measure both structures of $W$ and $Q$ via

<table>
<thead>
<tr>
<th>$2n$</th>
<th>50</th>
<th>100</th>
<th>150</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td></td>
<td>AA^* - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>UU^* - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>UUT - I</td>
<td></td>
<td>_F$</td>
</tr>
</tbody>
</table>

Table 4.19: Statistical data for real symplectic matrices and their polar factor
<table>
<thead>
<tr>
<th>2n</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50</td>
<td>100</td>
<td>150</td>
<td>200</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sweeps</td>
<td>7.25</td>
<td>8.05</td>
<td>8.33</td>
<td>8.73</td>
</tr>
<tr>
<td>reloff</td>
<td>4.55 x 10^{-14}</td>
<td>6.44 x 10^{-14}</td>
<td>5.71 x 10^{-14}</td>
<td>9.84 x 10^{-14}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AA^* - I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UU^T - I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HH^* - I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td>H - HT</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WW^* - I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WW^T - I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td>QQ^* - I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td>rel.singular</td>
<td>1.28 x 10^{-14}</td>
<td>2.54 x 10^{-14}</td>
<td>3.71 x 10^{-14}</td>
<td>4.83 x 10^{-14}</td>
</tr>
</tbody>
</table>

Table 4.20: Statistical data for the real symplectic SVD: \( S_{48} \)

<table>
<thead>
<tr>
<th>2n</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>7.15</td>
<td>8.10</td>
<td>8.88</td>
<td>9.13</td>
</tr>
<tr>
<td>reloff</td>
<td>1.30 x 10^{-14}</td>
<td>1.69 x 10^{-13}</td>
<td>7.76 x 10^{-14}</td>
<td>1.13 x 10^{-13}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AA^* - I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UU^T - I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HH^* - I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td>H - HT</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WW^* - I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WW^T - I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td>QQ^* - I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td>rel.singular</td>
<td>1.36 x 10^{-14}</td>
<td>2.86 x 10^{-14}</td>
<td>4.19 x 10^{-14}</td>
<td>5.56 x 10^{-14}</td>
</tr>
</tbody>
</table>

Table 4.21: Statistical data for the real symplectic SVD: \( S_{4rs} \)
### Table 4.22: Statistical data for the real symplectic SVD: $S_{ss}$

<table>
<thead>
<tr>
<th>2n</th>
<th>50</th>
<th>100</th>
<th>150</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>sweeps</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>4.25</td>
<td>4.70</td>
<td>5.00</td>
<td>5.12</td>
</tr>
<tr>
<td>reloff</td>
<td>$1.17 \times 10^{-13}$</td>
<td>$3.86 \times 10^{-14}$</td>
<td>$1.62 \times 10^{-14}$</td>
<td>$3.65 \times 10^{-14}$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>AA^* - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>UU^T - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>UU^T - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>HH^* - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>H - HT</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>WW^* - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>WW^T - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>QQ^* - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>QQ^T - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$2.55 \times 10^{-14}$</td>
<td>$5.23 \times 10^{-14}$</td>
<td>$8.40 \times 10^{-14}$</td>
<td>$1.05 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.Matlab</td>
<td>$1.90 \times 10^{-15}$</td>
<td>$2.61 \times 10^{-15}$</td>
<td>$3.17 \times 10^{-15}$</td>
<td>$3.58 \times 10^{-15}$</td>
</tr>
<tr>
<td>releig</td>
<td>$1.52 \times 10^{-14}$</td>
<td>$2.93 \times 10^{-14}$</td>
<td>$4.30 \times 10^{-14}$</td>
<td>$5.98 \times 10^{-14}$</td>
</tr>
<tr>
<td>rel.singular</td>
<td>$1.41 \times 10^{-14}$</td>
<td>$2.85 \times 10^{-14}$</td>
<td>$4.43 \times 10^{-14}$</td>
<td>$5.63 \times 10^{-14}$</td>
</tr>
</tbody>
</table>

### Table 4.23: Statistical data for the real symplectic SVD: $S_{srs}$

<table>
<thead>
<tr>
<th>2n</th>
<th>50</th>
<th>100</th>
<th>150</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>sweeps</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>4.23</td>
<td>4.70</td>
<td>5.00</td>
<td>5.40</td>
</tr>
<tr>
<td>reloff</td>
<td>$1.55 \times 10^{-14}$</td>
<td>$4.85 \times 10^{-14}$</td>
<td>$5.21 \times 10^{-14}$</td>
<td>$1.53 \times 10^{-14}$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>AA^* - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>UU^T - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>UU^T - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>HH^* - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>H - HT</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>WW^* - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>WW^T - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>QQ^* - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>QQ^T - I</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$3.01 \times 10^{-14}$</td>
<td>$6.06 \times 10^{-14}$</td>
<td>$1.05 \times 10^{-13}$</td>
<td>$1.33 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.Matlab</td>
<td>$1.90 \times 10^{-15}$</td>
<td>$2.61 \times 10^{-15}$</td>
<td>$3.17 \times 10^{-15}$</td>
<td>$3.58 \times 10^{-15}$</td>
</tr>
<tr>
<td>releig</td>
<td>$1.66 \times 10^{-14}$</td>
<td>$3.25 \times 10^{-14}$</td>
<td>$5.57 \times 10^{-14}$</td>
<td>$7.06 \times 10^{-14}$</td>
</tr>
<tr>
<td>rel.singular</td>
<td>$1.67 \times 10^{-14}$</td>
<td>$3.22 \times 10^{-14}$</td>
<td>$5.52 \times 10^{-14}$</td>
<td>$6.98 \times 10^{-14}$</td>
</tr>
<tr>
<td>2n</td>
<td>50</td>
<td>100</td>
<td>150</td>
<td>200</td>
</tr>
<tr>
<td>----</td>
<td>----------</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
</tr>
<tr>
<td></td>
<td>4.30</td>
<td>1.00 × 10^{-13}</td>
<td>1.48 × 10^{-14}</td>
<td>1.26 × 10^{-14}</td>
</tr>
<tr>
<td></td>
<td>5.02</td>
<td>1.43 × 10^{-14}</td>
<td>2.99 × 10^{-14}</td>
<td>3.38 × 10^{-14}</td>
</tr>
<tr>
<td></td>
<td>5.04</td>
<td>1.72 × 10^{-14}</td>
<td>4.23 × 10^{-14}</td>
<td>7.09 × 10^{-14}</td>
</tr>
<tr>
<td></td>
<td>5.43</td>
<td>3.44 × 10^{-14}</td>
<td>5.33 × 10^{-14}</td>
<td>1.12 × 10^{-13}</td>
</tr>
</tbody>
</table>

Table 4.24: Statistical data for the real symplectic SVD: \( S_{6n} \)

<table>
<thead>
<tr>
<th>2n</th>
<th>50</th>
<th>100</th>
<th>150</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4.06</td>
<td>1.35 × 10^{-14}</td>
<td>1.48 × 10^{-14}</td>
<td>1.26 × 10^{-14}</td>
</tr>
<tr>
<td></td>
<td>4.80</td>
<td>1.38 × 10^{-13}</td>
<td>2.99 × 10^{-14}</td>
<td>3.38 × 10^{-14}</td>
</tr>
<tr>
<td></td>
<td>5.02</td>
<td>3.14 × 10^{-14}</td>
<td>4.23 × 10^{-14}</td>
<td>7.09 × 10^{-14}</td>
</tr>
<tr>
<td></td>
<td>5.20</td>
<td>2.19 × 10^{-14}</td>
<td>5.33 × 10^{-14}</td>
<td>1.12 × 10^{-13}</td>
</tr>
</tbody>
</table>

Table 4.25: Statistical data for the real symplectic SVD: \( S_{6_{rs}} \)
\[ \|WW^* - I\|_F, \|WW^T - I\|_F, \text{ and } \|QQ^* - I\|_F, \|QQ^T - I\|_F, \] respectively. We see that all strategies performed well to the order of $10^{-13}$ in computing the real symplectic orthogonal matrices $W$ and $Q$ for all dimensions under consideration.

- The row labeled $\text{diag.alg}$ reports the deviation of $\Sigma$ from being real symplectic diagonal matrix. Here $\text{diag.alg} = \max | \text{diag}(S_1) \ast \text{diag}(S_2) - 1 |$ where $S_1 = \Sigma(1 : n, 1 : n)$, $S_2 = \Sigma(n + 1 : 2n, n + 1 : 2n)$, and $\ast$ denotes array multiplication (as in MATLAB). Note that all singular values of $A$ are all real, and occur in pairs $(\sigma, 1/\sigma)$. Here $\text{diag.Matlab}$ measures the pairing structure in the singular values computed by MATLAB’s $\text{svd}$ function. Let $v$ be the vector of the sorted singular values computed by $\text{svd}(A)$. Then $\text{diag.Matlab} = \max | x \ast y - 1 |$ where $x = v(1 : n)$ and $y = v(2n : -1 : n+1)$. As we see from Tables 4.20 - 4.25, the loss of the pairing structure in the singular values computed by our algorithms was of the order $10^{-13}$ while the loss of pairing structure of singular values computed by MATLAB was of the order $10^{-15}$.

- The last two rows of Tables 4.20 - 4.25 report the maximum relative error in the computed eigenvalues of $H$ and the maximum relative error in the computed singular values of $A$, respectively. Theoretically, the eigenvalues of $H$ and the singular values of $A$ are the same. To measure these maximum relative errors, we sorted the computed eigenvalues of $H$ and the computed singular values of $A$ and compared them with the sorted eigenvalues obtained by MATLAB’s $\text{eig}$ function and the sorted singular values obtained by MATLAB’s $\text{svd}$ function, respectively. The row $\text{releig}$ shows the maximum relative error in the computed eigenvalues as given in (4.26) and the row $\text{rel.singular}$ reports the maximum relative error

\[
\text{rel.singular} = \max_j \left| \frac{\sigma_j^{\text{mat}} - \sigma_j^{\text{alg}}}{\sigma_j^{\text{mat}}} \right|, \tag{4.28}
\]

where $\{\sigma_j^{\text{mat}}\}$ and $\{\sigma_j^{\text{alg}}\}$ denote the singular values computed by MATLAB’s $\text{svd}$ function and by our strategies, respectively. We see that the eigenvalues and the singular values computed by our strategies were comparable to MATLAB up to order $10^{-14}$.

We finish this section by showing the accuracy of computing the singular values by our six strategies, $S_{6s}$, $S_{6rs}$, $S_{8s}$, $S_{8rs}$, $S_{6s}$, and $S_{6rs}$. To investigate this, the experiments were performed by applying all strategies to a real symplectic test matrix with pre-chosen singular values. Then we compared the chosen singular values to singular values computed by our strategies and to those computed by MATLAB’s $\text{svd}$ function. Some of the experiments are shown below.
Example 5. A real symplectic matrix $A$ was generated with singular values $10, 5, 2, \frac{1}{5}, \frac{1}{10}$. In this case the given matrix $A$ is a well-conditioned matrix having distinct singular values with condition number 100. Table 4.26 shows the exact singular values and the relative errors in the singular values approximations computed by our algorithms and by MATLAB's `svd` function.

<table>
<thead>
<tr>
<th>$\sigma_j$</th>
<th>$S_{4\sigma}$</th>
<th>$S_{4\tau}$</th>
<th>$S_{5\sigma}$</th>
<th>$S_{5\tau}$</th>
<th>$S_{6\sigma}$</th>
<th>$S_{6\tau}$</th>
<th>Matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/10</td>
<td>5.6e-16</td>
<td>4.2e-16</td>
<td>3.3e-15</td>
<td>1.7e-15</td>
<td>2.1e-15</td>
<td>4.2e-16</td>
<td>8.3e-16</td>
</tr>
<tr>
<td>1/5</td>
<td>1.4e-15</td>
<td>3.2e-15</td>
<td>1.1e-15</td>
<td>9.9e-16</td>
<td>1.4e-15</td>
<td>1.7e-15</td>
<td>3.3e-15</td>
</tr>
<tr>
<td>1/2</td>
<td>2.2e-16</td>
<td>4.4e-16</td>
<td>1.4e-15</td>
<td>2.2e-16</td>
<td>0</td>
<td>2.2e-16</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3.3e-16</td>
<td>4.4e-16</td>
<td>2.2e-16</td>
<td>4.4e-16</td>
<td>0</td>
<td>1.1e-15</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1.1e-15</td>
<td>7.1e-16</td>
<td>8.9e-16</td>
<td>2.5e-15</td>
<td>1.2e-15</td>
<td>5.3e-16</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>1.8e-16</td>
<td>1.2e-15</td>
<td>2.1e-15</td>
<td>2.3e-15</td>
<td>2.0e-15</td>
<td>7.1e-16</td>
</tr>
</tbody>
</table>

Table 4.26: Accuracy of the singular values $10, 5, 2, \frac{1}{5}, \frac{1}{10}$

Example 6. A real symplectic matrix $A$ was generated with singular values $a, b, c, \frac{1}{a}, \frac{1}{b}, \frac{1}{c}$ where $[a, b, c] = [3, 3, 3] + [9.6, 3.5, 2.1] \times 10^{-15}$. Here $A$ is a well-conditioned matrix and has two clusters of singular values of “multiplicity” 3 — singular values near 3 and near $\frac{1}{3}$. The exact singular values and the relative errors are given in Table 4.27.

<table>
<thead>
<tr>
<th>$\sigma_j$</th>
<th>$S_{4\sigma}$</th>
<th>$S_{4\tau}$</th>
<th>$S_{5\sigma}$</th>
<th>$S_{5\tau}$</th>
<th>$S_{6\sigma}$</th>
<th>$S_{6\tau}$</th>
<th>Matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/a</td>
<td>6.7e-16</td>
<td>6.7e-16</td>
<td>8.3e-16</td>
<td>5.0e-16</td>
<td>1.6e-16</td>
<td>5.0e-16</td>
<td>0</td>
</tr>
<tr>
<td>1/b</td>
<td>5.0e-16</td>
<td>0</td>
<td>0</td>
<td>9.9e-16</td>
<td>1.6e-16</td>
<td>9.9e-16</td>
<td>1.3e-15</td>
</tr>
<tr>
<td>1/c</td>
<td>6.7e-16</td>
<td>0</td>
<td>1.6e-16</td>
<td>9.9e-16</td>
<td>6.0e-16</td>
<td>9.9e-16</td>
<td>5.0e-16</td>
</tr>
<tr>
<td>c</td>
<td>1.5e-16</td>
<td>0</td>
<td>4.4e-16</td>
<td>1.3e-15</td>
<td>4.4e-16</td>
<td>1.3e-15</td>
<td>1.4e-16</td>
</tr>
<tr>
<td>b</td>
<td>1.5e-16</td>
<td>1.5e-16</td>
<td>5.9e-16</td>
<td>4.4e-16</td>
<td>1.1e-15</td>
<td>4.4e-16</td>
<td>8.8e-16</td>
</tr>
<tr>
<td>a</td>
<td>1.5e-16</td>
<td>1.5e-16</td>
<td>4.4e-16</td>
<td>1.1e-15</td>
<td>0</td>
<td>1.3e-16</td>
<td>2.9e-16</td>
</tr>
</tbody>
</table>

Table 4.27: Accuracy of the singular values near 3 and $\frac{1}{3}$

Example 7. A real symplectic matrix $A$ was generated with singular values $10^4, 10^2, 10, 10^{-1}, 10^{-2}, 10^{-4}$. In this case, $A$ is an ill-conditioned matrix with condition number $10^8$. The exact singular values and the relative errors are reported in Table 4.28.

Example 8. A real $20 \times 20$ symplectic matrix $A$ was generated with singular values $2, \frac{3}{2}, 4, \frac{4}{2}, 6, \frac{1}{5}, 7, \frac{7}{1}, a, b, c, \frac{1}{a}, \frac{1}{b}, \frac{1}{c}, x, y, z, \frac{1}{x}, \frac{1}{y}, \frac{1}{z}$ where $[a, b, c] = [3, 3, 3] + [2.2, 3.1, 7.1] \times 10^{-15}$ and $[x, y, z] = [5, 5, 5] + [1.8, 5.3, 8.9] \times 10^{-15}$. In this case, $A$ is well-conditioned and has 4 clusters of singular values of “multiplicity” 3 —
Table 4.28: Accuracy of the singular values $10^4, 10^2, 10, \frac{1}{10}, \frac{1}{10^2}, \frac{1}{10^4}$

<table>
<thead>
<tr>
<th>$\sigma_j$</th>
<th>$S_{4s}$</th>
<th>$S_{4rs}$</th>
<th>$S_{6s}$</th>
<th>$S_{6rs}$</th>
<th>$S_{6s}$</th>
<th>$S_{6rs}$</th>
<th>Matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1/10^4$</td>
<td>5.2e-9</td>
<td>6.9e-9</td>
<td>2.7e-9</td>
<td>1.8e-9</td>
<td>2.8e-9</td>
<td>1.9e-9</td>
<td>1.7e-9</td>
</tr>
<tr>
<td>$1/10^2$</td>
<td>1.7e-12</td>
<td>4.7e-12</td>
<td>8.9e-12</td>
<td>6.3e-11</td>
<td>1.7e-11</td>
<td>6.2e-11</td>
<td>3.7e-11</td>
</tr>
<tr>
<td>1/10</td>
<td>2.8e-13</td>
<td>8.8e-13</td>
<td>1.0e-12</td>
<td>6.3e-14</td>
<td>2.5e-12</td>
<td>2.1e-12</td>
<td>2.9e-12</td>
</tr>
<tr>
<td>10</td>
<td>1.9e-14</td>
<td>2.3e-14</td>
<td>1.2e-13</td>
<td>5.7e-14</td>
<td>2.7e-13</td>
<td>9.2e-14</td>
<td>3.2e-14</td>
</tr>
<tr>
<td>$10^2$</td>
<td>1.3e-15</td>
<td>5.7e-16</td>
<td>1.7e-15</td>
<td>4.2e-16</td>
<td>3.1e-15</td>
<td>2.8e-16</td>
<td>7.1e-16</td>
</tr>
<tr>
<td>$10^4$</td>
<td>7.3e-16</td>
<td>0</td>
<td>3.6e-16</td>
<td>7.2e-16</td>
<td>7.2e-16</td>
<td>1.6e-15</td>
<td>3.6e-16</td>
</tr>
</tbody>
</table>

Table 4.29: Accuracy of the singular values near $3, \frac{1}{3}, 5$ and $\frac{1}{5}$

<table>
<thead>
<tr>
<th>$\sigma_j$</th>
<th>$S_{4s}$</th>
<th>$S_{4rs}$</th>
<th>$S_{6s}$</th>
<th>$S_{6rs}$</th>
<th>$S_{6s}$</th>
<th>$S_{6rs}$</th>
<th>Matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>8.9e-16</td>
<td>1.5e-15</td>
<td>1.4e-15</td>
<td>7.4e-16</td>
<td>2.8e-15</td>
<td>1.4e-16</td>
<td>4.4e-16</td>
</tr>
<tr>
<td>b</td>
<td>5.9e-16</td>
<td>3.0e-16</td>
<td>0</td>
<td>1.0e-15</td>
<td>3.1e-15</td>
<td>0</td>
<td>4.4e-16</td>
</tr>
<tr>
<td>c</td>
<td>0</td>
<td>1.8e-15</td>
<td>1.3e-15</td>
<td>3.2e-15</td>
<td>3.1e-15</td>
<td>1.7e-15</td>
<td>7.4e-16</td>
</tr>
<tr>
<td>1/a</td>
<td>1.5e-15</td>
<td>1.3e-15</td>
<td>1.1e-15</td>
<td>3.1e-15</td>
<td>1.8e-15</td>
<td>2.1e-15</td>
<td>9.9e-16</td>
</tr>
<tr>
<td>1/b</td>
<td>3.3e-16</td>
<td>1.7e-15</td>
<td>8.3e-16</td>
<td>3.0e-15</td>
<td>8.3e-16</td>
<td>2.3e-15</td>
<td>8.3e-16</td>
</tr>
<tr>
<td>1/c</td>
<td>5.0e-16</td>
<td>3.3e-16</td>
<td>5.9e-16</td>
<td>3.5e-15</td>
<td>9.9e-16</td>
<td>3.1e-15</td>
<td>2.0e-15</td>
</tr>
<tr>
<td>x</td>
<td>8.9e-16</td>
<td>1.8e-16</td>
<td>1.1e-15</td>
<td>1.2e-15</td>
<td>1.7e-16</td>
<td>0</td>
<td>1.7e-16</td>
</tr>
<tr>
<td>y</td>
<td>1.1e-15</td>
<td>7.1e-16</td>
<td>7.1e-16</td>
<td>7.1e-16</td>
<td>8.8e-16</td>
<td>3.2e-15</td>
<td>7.1e-16</td>
</tr>
<tr>
<td>z</td>
<td>2.0e-15</td>
<td>4.1e-15</td>
<td>4.2e-15</td>
<td>2.4e-15</td>
<td>1.9e-15</td>
<td>3.3e-15</td>
<td>7.1e-16</td>
</tr>
<tr>
<td>1/x</td>
<td>1.8e-15</td>
<td>3.2e-15</td>
<td>0</td>
<td>5.5e-16</td>
<td>1.2e-15</td>
<td>2.2e-15</td>
<td>2.3e-15</td>
</tr>
<tr>
<td>1/y</td>
<td>2.4e-15</td>
<td>1.4e-16</td>
<td>2.7e-16</td>
<td>9.7e-16</td>
<td>2.3e-15</td>
<td>1.3e-16</td>
<td>2.7e-16</td>
</tr>
<tr>
<td>1/z</td>
<td>5.6e-16</td>
<td>5.6e-16</td>
<td>3.3e-15</td>
<td>1.3e-16</td>
<td>5.2e-15</td>
<td>9.7e-16</td>
<td>3.1e-15</td>
</tr>
</tbody>
</table>

The results from all four examples as shown in Tables 4.26-4.29 report that the singular values computed by each strategy have the same accuracy. When $A$ is well-conditioned as in Examples 5, 6, and 8 our strategies give the singular values with the relative error of order $\varepsilon$ as same as does by MATLAB. Moreover, from Table 4.28 the loss of accuracy of the small singular value computed by our strategies and by MATLAB is obvious. Therefore, we can conclude that the singular values computed by our strategies and by MATLAB’s $svd$ function are comparable.
Chapter 5

Computing the Real Symplectic SVD: One-sided Jacobi

In this Chapter we develop an adaptation of the one-sided Jacobi method to compute a real symplectic SVD. Note that in contrast to previous chapters where the matrix had double structure — real symplectic and symmetric — we are now working on a matrix that is just real symplectic. We begin by giving a description of the basic idea of the one-sided Jacobi method.

5.1 General One-sided Jacobi Method

The one-sided Jacobi method is one of methods used to compute an SVD of a matrix; it was introduced by Hestenes [14], and has been the topic of numerous publications (see [6], [34], [38], [43], for example). In general the one-sided Jacobi method can be applied to an arbitrary matrix of dimension $m \times n$. Our presentation here deals with a real $n \times n$ matrix $A$ with rank$(A) = n$. This method is mathematically equivalent to applying the Jacobi method to the symmetric matrix $A^TA$. But instead of repeatedly applying orthogonal similarity transformations as is done in the Jacobi method, we only compute an orthogonal matrix $V$ such that

$$AV = B,$$  \hspace{1cm} (5.1)

where the columns of $B$ are orthogonal.

The outline of the one-sided Jacobi algorithm is as follows. Setting $A = B^{(0)}$, the method produces a sequence of matrices $\{B^{(\ell)}\}$, where

$$B^{(\ell)} = B^{(\ell-1)}R_{pq}, \quad \ell = 1, 2, \ldots; \quad p \neq q,$$  \hspace{1cm} (5.2)

and $R_{pq}$ is an $n \times n$ plane rotation such that two columns of $B^{(\ell)}$ become orthogonal. More specifically, for every iteration $\ell$ there is a corresponding index pair $(p, q)$ satisfying $1 \leq p < q \leq n$ and an angle $\theta$ such that $R_{pq}$ is of the form as in
Here $\theta$ is determined so that the columns $p$ and $q$ of $B^{(\ell)}$ are orthogonal to each other. In other words, $R_{pq}$ is chosen so that two off-diagonal elements $(p,q)$ and $(q,p)$ of the symmetric matrix $B^{(\ell)^T}B^{(\ell)}$ are made zero. By choosing the index $(p,q)$ and an angle $\theta$ appropriately in each step, the sequence $\{B^{(\ell)}\}$ converges to a real $n \times n$ matrix $B$ which has the property that $B^TB$ is a diagonal matrix consisting of the square of the singular values of $A$. The matrix $V$ can be now generated as a product of all rotations $R_{pq}$. So $V$ is orthogonal and we have (5.1) as desired.

An SVD of $A$ can be obtained from (5.1) as follows. Writing $B$ and $V$ in terms of their columns

$$B = [b_1, b_2, \ldots, b_n]; \quad V = [v_1, v_2, \ldots, v_n],$$

where $b_i$ and $v_i$ are $n$-vectors, and defining

$$\sigma_i = \|b_i\|, \quad i = 1, 2, \ldots, n,$$

$$u_i = \frac{b_i}{\sigma_i}, \quad i = 1, 2, \ldots, n,$$

we have

$$A = U \Sigma V^T,$$

where $U = [u_1, u_2, \ldots, u_n], V = [v_1, v_2, \ldots, v_n]$, and $\Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$. This is an SVD of $A$.

To determine a plane rotation $R_{pq}$ introduced in (5.2), we first write the matrix $B^{(\ell-1)}$ in terms of its columns: $B^{(\ell-1)} = [b_1^{(\ell-1)}, b_2^{(\ell-1)}, \ldots, b_n^{(\ell-1)}]$. Since right multiplication of $B^{(\ell-1)}$ by $R_{pq}$ affects only columns $p$ and $q$ of $B^{(\ell-1)}$, one obtains

$$b_p^{(\ell)} = c b_p^{(\ell-1)} - s b_q^{(\ell-1)},$$

$$b_q^{(\ell)} = s b_p^{(\ell-1)} + c b_q^{(\ell-1)}.$$

Note that choosing $\theta$ so that the columns $p$ and $q$ of $B^{(\ell)}$ are orthogonal also means setting the off-diagonal elements $(p,q)$ and $(q,p)$ in $B^{(\ell)^T}B^{(\ell)}$ (which are equal, by symmetry) to zero. That is, if we let $x_{pp} = (B^{(\ell-1)^T}B^{(\ell-1)})_{pp}, x_{qq} = (B^{(\ell-1)^T}B^{(\ell-1)})_{qq}$, and $x_{pq} = (B^{(\ell-1)^T}B^{(\ell-1)})_{pq}$, then setting the off-diagonal elements $(p,q)$ and $(q,p)$ in $B^{(\ell)^T}B^{(\ell)}$ to zero yields

$$(x_{pp} - x_{qq})cs + x_{pq}(c^2 - s^2) = 0.$$ 

This equation is analogous to (2.9). Hence $(c,s)$ can be obtained as in (2.11).

Moreover, choosing either the small angle or the large angle makes a difference in the order of $b_p^{(\ell)}$ and $b_q^{(\ell)}$, that is, if $\theta_{\text{small}}$ is chosen, then the order of $\|b_p^{(\ell)}\|^2$ and $\|b_q^{(\ell)}\|^2$ is the same as that of $\|b_p^{(\ell-1)}\|^2$ and $\|b_q^{(\ell-1)}\|^2$ [34]. As was mentioned in [6], choosing the large angle whenever $x_{pp} < x_{qq}$ in each iteration leads to

$$\|b_1\|^2 \geq \|b_2\|^2 \geq \cdots \geq \|b_n\|^2.$$
When this happens, the diagonal elements of $\Sigma$ are in decreasing order, i.e. $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$.

To complete the one-sided Jacobi method, a strategy for selecting the rotation index pair $(p, q)$ is required. Since computing an SVD of $A$ by the one-sided Jacobi method is the same as solving the eigenproblem of $A^T A$ by the Jacobi method, the strategy for selecting the rotation index pair can be chosen from many available strategies for the Jacobi method. For more discussion about the one-sided Jacobi method, see, for example, [6], [31], [34], [38], [43], [44].

### 5.2 Adaptation to Real Symplectic Matrices

In our work, the given matrix $A$ is a real $2n \times 2n$ symplectic matrix. By Theorem 1.5, a real symplectic SVD of $A$ exists. That is,

$$A = U \Sigma V^T,$$

where $U$ and $V$ are real symplectic orthogonal and $\Sigma$ is real symplectic diagonal of the form

$$\Sigma = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_1^{-1} \end{bmatrix}, \text{ where } \Sigma_1 = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n).$$

To calculate a real symplectic SVD of $A$ by using a one-sided Jacobi method that preserves the real symplectic structure of $A$, our tools must be limited to matrices from $SpO(2n, \mathbb{R})$ as given in (1.12).

Since a one-sided Jacobi method for the SVD of $A$ implicitly acts on $A^T A$ by a two-sided action, the tools in Chapter 2 — real $2n \times 2n$ symplectic plane rotation as in (2.17), real $2n \times 2n$ symplectic double rotations with direct sum embedding as in (2.32), and real $2n \times 2n$ symplectic double rotations with concentric embedding as in (2.37) — can be used. For convenience in accordance with the one-sided Jacobi method, we will refer to these tools as one-sided rotations, one-sided double rotations, and one-sided concentric rotations, respectively. Algorithms for computing these tools are given at the end of this section.

The sequence of pivots for our one-sided Jacobi-like algorithms is chosen from sweep designs $S_4$, $S_5$, and $S_6$.

The criterion for setting the off-diagonal elements $(p, q)$ and $(q, p)$ of $X = [x_{ij}] = A^T A$ to zero is defined by

$$\frac{|x_{pq}|}{\sqrt{x_{pp} x_{qq}}} > \text{threshold},$$

where threshold is defined by the user. Here we set threshold $= 2n \times \text{eps}$ in our algorithms. That is, the orthogonalization of the column pair $(p, q)$ is skipped.
when the reverse inequality in (5.4) holds. The criterion given in (5.4) has been suggested, for example, in [5], [34], [38], [41].

Note that there are no repetitions of pivot elements in the sweep designs $S_4$, $S_5$ and $S_6$. Recall that the number of iterations in a full sweep of $S_4$ is $n^2$ while the number of iterations in a full sweep of $S_5$ and $S_6$ is $n(2n-1)$. Algorithm with sweep design $S_4$ is set to terminate when $n^2$ consecutive pivot elements (not necessarily in one sweep) are skipped while algorithms with sweep designs $S_5$ and $S_6$ are terminated when $2n(n-1)$ consecutive pivot elements are skipped. Equivalently, our algorithms stop when one “sweep” has been skipped.

We now present the one-sided Jacobi-like algorithms for computing a real symplectic SVD of a real symplectic matrix $A$. We will use $OS_4$, $OS_5$ and $OS_6$ to denote the one-sided Jacobi algorithms using the small angle and strategies $S_4$, $S_5$ and $S_6$, respectively. $OS_{4s}$, $OS_{5s}$ and $OS_{6s}$ will denote the one-sided Jacobi algorithms using strategies $S_4$, $S_5$ and $S_6$ with the sorting angle, respectively.

Pseudo-code for each algorithm is given below. A Boolean parameter “sort” is given to each algorithm. This parameter will determine the use of either the small angle or the sorting angle. If sort is zero, then the small angle is used by each function called by the algorithm. On the other hand, when the input parameter sort $\neq 0$, the sorting angle applies to an appropriate iteration. That is, for the algorithm $OS_{4s}$, the sorting angle is used at every iteration of every sweep as described in Section 4.2.4. In contrast, for the algorithms $OS_{5s}$ and $OS_{6s}$, the sorting angle is used when rotation index pairs $(p,q)$ are in the strict upper triangular part of $E$ and in upper off-diagonal block $F$ where $A = [E \ F]$. The use of the sorting angle results in the first $n$ diagonal entries of $E$ appearing, at convergence, in decreasing order on the diagonal. The parameter “count” keeps track of the number of iterations which are skipped. Pseudo-code for the functions One-sided Rotation, One-sided Double, and One-sided Concentric used by Algorithms 16 and 18 is given later at the end of this section.

**Algorithm 10 ($OS_4$, $OS_{4s}$).** Given a symplectic matrix $A \in \mathbb{R}^{2n \times 2n}$, a Boolean parameter “sort”, and tol $> 0$, this algorithm computes the singular values $\sigma_i$, and real symplectic orthogonal matrices $U$ and $V$ so that $A = U\Sigma V^T$ where $\Sigma = \text{diag}(\sigma_i)$ is also symplectic. If “sort” is zero, then the first $n$ diagonal entries of $\Sigma$ do not appear in any particular order on the diagonal. On the other hand, if “sort” is nonzero, then the first $n$ diagonal entries of $\Sigma$ appearing in decreasing order.

$V = I_{2n}$

while count $< n^2$

for $i = 1 : n$

$Q = \text{One-sided Rotation}(A, i, \text{sort}, \text{tol})$

$A = AQ; V = VQ$

endfor
for i = 1 : n - 1
    for j = i + 1 : n
        Q = One-side Double(A, i, j, sort, tol)
        A = AQ; V = VQ
    endfor
endfor
for i = 1 : n - i
    for j = n + i + 1 : 2n
        Q = One-sided Concentric(A, i, j, sort, tol)
        A = AQ; V = VQ
    endfor
endfor
endwhile
for i = 1 : 2n
    σ_i = ||A(:,i)||; U(:,i) = \frac{A(:,i)}{σ_i}
endfor

Algorithm 11 (OS_S, OS_S5). Given a symplectic matrix A ∈ \mathbb{R}^{2n×2n}, a Boolean parameter “sort”, and tol > 0, this algorithm computes the singular values σ_i, and real symplectic orthogonal matrices U and V so that A = UΣV^T where Σ = \text{diag}(σ_i) is also symplectic. If “sort” is zero, then the first n diagonal entries of Σ do not appear in any particular order on the diagonal. On the other hand, if “sort” is nonzero, then the first n diagonal entries of Σ appearing in decreasing order.

V = I_{2n}
while count < n(2n - 1)
    for i = 1 : n
        for j = i + 1 : 2n
            if j ≤ n
                Q = One-sided Double(A, i, j, sort, tol)
                A = AQ; V = VQ
            elseif j = n + i
                Q = One-sided Rotation(A, i, sort, tol)
                A = AQ; V = VQ
            else
                Q = One-sided Concentric(A, i, j, sort, tol)
                A = AQ; V = VQ
            endif
        endfor
    endfor
endwhile

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
for $i = n + 1 : 2n - 1$
  for $j = i + 1 : 2n$
    $Q = \text{One-sided Double}(A, i, j, 0, \text{tol})$
    $A = AQ; V = VQ$
  endfor
endfor
endwhile

for $i = 1 : 2n$
  $\sigma_i = \|A(:,i)\|; U(:,i) = \frac{A(:,i)}{\sigma_i}$
endfor

**Algorithm 12 ($\mathcal{OS}_6, \mathcal{OS}_{6a}$).** Given a symplectic matrix $A \in \mathbb{R}^{2n \times 2n}$, a Boolean parameter "sort", and $\text{tol} > 0$, this algorithm computes the singular values $\sigma_i$, and real symplectic orthogonal matrices $U$ and $V$ so that $A = U\Sigma V^T$ where $\Sigma = \text{diag}(\sigma_i)$ is also symplectic. If "sort" is zero, then the first $n$ diagonal entries of $\Sigma$ do not appear in any particular order on the diagonal. On the other hand, if "sort" is nonzero, then the first $n$ diagonal entries of $\Sigma$ appearing in decreasing order.

$V = I_{2n}$
while count $< n(2n - 1)$
  for $i = 1 : n - 1$
    for $j = i + 1 : n$
      $Q = \text{One-sided Double}(A, i, j, \text{sort}, \text{tol})$
      $A = AQ; V = VQ$
    endfor
  endfor
  for $i = 1 : n$
    $Q = \text{One-sided Rotation}(A, i, \text{sort}, \text{tol})$
    $A = AQ; V = VQ$
  endfor
  for $i = 1 : n - 1$
    for $j = n + i + 1 : 2n$
      $Q = \text{One-sided Concentric}(A, i, j, \text{sort}, \text{tol})$
      $A = AQ; V = VQ$
    endfor
  endfor
  for $j = n + 1 : 2n - 1$
    for $i = j - n + 1 : n$
      $Q = \text{One-sided Concentric}(A, i, j, \text{sort}, \text{tol})$
      $A = AQ; V = VQ$
  endfor
endwhile

99
endfor
endfor
for $i = n + 1 : 2n - 1$
    for $j = i + 1 : 2n$
        $Q = \text{One-sided Double}(A, i, j, 0, tol)$
        $A = AQ; \ V = VQ$
    endfor
endfor
endwhile
for $i = 1 : 2n$
    $\sigma_i = \|A(:,i)\|_1; \ U(:,i) = \frac{A(:,i)}{\sigma_i}$
endfor

We now finish this section by giving pseudo-code for the functions called by Algorithms 16 and 18. These functions differ from those described in Sections 2.2.2, 2.3.1 - 2.3.2 in Chapter 2 only in that they first need to compute certain elements of $A^T A$. For example, function Rotation in Section 2.2.2 and function One-sided Rotation given below both need the elements in positions $(p, p)$, $(n + p, n + p)$ and $(p, n + p)$. Function Rotation needs those elements of $A$ while the function One-sided Rotation needs those elements of $A^T A$. 

**function: $Q = \text{One-sided Rotation}(A, p, \text{sort}, tol)$**

Given a real matrix $A \in \mathbb{R}^{2n \times 2n}$; an integer $p$ that satisfies $1 < p < n$, a Boolean parameter "sort"; and $\text{tol} > 0$, this algorithm computes a real symplectic plane rotation $Q \in \mathbb{R}^{2n \times 2n}$ as in (2.17) such that the columns $p$ and $n + p$ of $AQ$ are orthogonal to each other. That is, if $X = A^T A$, then the two off-diagonal entries in positions $(p, n + p)$ and $(n + p, p)$ of $\tilde{X} = Q^T A^T AQ$ are now zero. If $\text{sort} = 0$, then the order of the entries in positions $(p, p)$ and $(n + p, n + p)$ of $X$ and $\tilde{X}$ is the same. If $\text{sort}$ is nonzero, then the sorting angle is used for ordering the entries in positions $(p, p)$ and $(n + p, n + p)$ of $\tilde{X}$ in decreasing order.

- Compute the entries $(p, p)$, $(p, n + p)$, and $(n + p, n + p)$ of $A^T A$
- $x_{pp} = \|A(:,p)\|^2; \ x_{n+p,n+p} = \|A(:,n+p)\|^2; \ x_{p,n+p} = (A(:,p))^T A(:,n+p))$
  - if $|x_{p,n+p}| > \text{tol}$ then % compute small angle
    - $\tau = \frac{x_{p,n+p} - x_{pp}}{2x_{n+p,n+p}}$
    - $\theta = \text{sign}(\tau) \frac{|\tau| + \sqrt{\tau^2 + 1}}{2}$ % as in (2.5)
    - if (sort $\neq 0$) and $(x_{pp} < x_{n+p,n+p})$ then
      - $t = -\frac{1}{i}$ % use large angle

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
endif
\[ c = \frac{1}{\sqrt{t^2 + 1}} \quad ; \quad s = tc \]
else
\[ c = 1 \quad ; \quad s = 0 \]
endif
\[ Q = I_{2n} \quad ; \quad Q(p,p) = c \quad ; \quad Q(p,n+p) = s \quad ; \quad Q(n+p,p) = -s \quad ; \quad Q(n+p,n+p) = c \]

**function: Q = One-sided Double**(\( A, p, q, sort, tol \))

Given a real matrix \( A \in \mathbb{R}^{2n \times 2n} \), integers \( p \) and \( q \) that satisfy \( 1 \leq p < q \leq n \), a Boolean parameter “sort”, and \( tol > 0 \), this algorithm computes a real symplectic double rotation with direct sum embedding \( Q \in \mathbb{R}^{2n \times 2n} \) as in (2.32) such that the columns \( p \) and \( q \) of \( AQ \) are orthogonal to each other. That is, if \( X = AT^A \), then the two off-diagonal entries in positions \( (p,q) \) and \( (q,p) \) of \( \tilde{X} = Q^T AT AQ \) are now zero. If sort = 0, then the order of the entries in positions \( (p,p) \) and \( (q,q) \) of \( X \) and \( \tilde{X} \) is the same. If sort is nonzero, then the sorting angle is used for ordering the entries in positions \( (p,p) \) and \( (q,q) \) of \( \tilde{X} \) in decreasing order.

\[
\begin{align*}
\% \text{Compute the entries } (p,p), (p,q), \text{ and } (q,q) \text{ of } AT^A \\
x_{pp} = \|A(:,p)\|^2; \quad x_{qq} = \|A(:,q)\|^2; \quad x_{pq} = (A(:,p))^T(A(:,q)) \\
\text{if } |x_{pq}| > tol \text{ then} \% \text{compute small angle} \\
\tau &= \frac{x_{qq} - x_{pp}}{2x_{pq}} \\
t &= \frac{\text{sign}(\tau)}{|\tau| + \sqrt{\tau^2 + 1}} \% \text{as in (2.5)} \\
\text{if } (\text{sort} \neq 0) \text{ and } (x_{pp} < x_{qq}) \text{ then} \\
\quad t &= -\frac{1}{t} \% \text{use large angle} \\
\text{endif} \\
\quad c &= \frac{1}{\sqrt{t^2 + 1}} \quad ; \quad s = tc \\
\text{else} \\
\quad c &= 1 \quad ; \quad s = 0 \\
\text{endif} \\
Q = I_{2n} \quad ; \quad Q(p,p) = c \quad ; \quad Q(p,q) = s \quad ; \quad Q(q,p) = -s \quad ; \quad Q(q,q) = c \\
Q(n+p,n+p) = c \quad ; \quad Q(n+p,n+q) = s \quad ; \quad Q(n+q,n+p) = -s \quad ; \quad Q(n+q,n+q) = c
\]

Note that an algorithm for orthogonalizing columns \( n + p \) and \( n + q \) can be obtained by replacing \( p \) and \( q \) with \( n + p \) and \( n + q \), respectively in function One-sided Double.

**function: Q = One-sided Concentric**(\( A, p, n+q, sort, tol \))
Given a real matrix $A \in \mathbb{R}^{2n \times 2n}$, and integers $p$ and $n + q$ that satisfy $1 \leq p < q \leq n$, a Boolean parameter "sort", and $\text{tol} > 0$, this algorithm computes a real symplectic double rotation with concentric embedding $Q \in \mathbb{R}^{2n \times 2n}$ as in (2.37) such that the columns $p$ and $n + q$ of $AQ$ are orthogonal to each other. That is, if $X = A^T A$, then the two off-diagonal entries in position $(p, n + q)$ and $(n + q, p)$ of $\tilde{X} = Q^T A^T A Q$ are now zero. If sort $= 0$, then the order of the entries in positions $(p, p)$ and $(n + q, n + q)$ of $X$ and $\tilde{X}$ is the same. If sort is nonzero, then the sorting angle is used for ordering the entries in positions $(p, p)$ and $(n + q, n + q)$ of $\tilde{X}$ in decreasing order.

% Compute the entries $(p, p), (p, n + q),$ and $(n + q, n + q)$ of $A^T A$
\[
x_{pp} = \|A(:, p)\|^2; \quad x_{n+q, n+q} = \|A(:, n + q)\|^2; \quad x_{p, n+q} = (A(:, p))^T(A(:, n + q))
\]
if $|x_{p, n+q}| > \text{tol}$ then % compute small angle
\[
\tau = \frac{x_{n+q, n+q} - x_{pp}}{2x_{p, n+q}} \\
t = \frac{\text{sign } \tau}{|\tau| + \sqrt{\tau^2 + 1}} \quad \% \text{as in (2.5)}
\]
if (sort $\neq 0$ and $b_{pp} < b_{n+q, n+q}$) then
\[
t = -\frac{1}{t} \quad \% \text{use large angle}
\]
endif
\[
c = \frac{1}{\sqrt{t^2 + 1}}; \quad s = tc
\]
else
\[
c = 1; \quad s = 0
\]
endif
\[
Q = I_{2n}; \quad Q(p, p) = c; \quad Q(p, n + q) = s; \quad Q(n + q, p) = -s; \quad Q(n + q, n + q) = c; \\
Q(qq) = c; \quad Q(q, n + p) = s; \quad Q(n + p, q) = -s; \quad Q(n + p, n + p) = c
\]

Note that an algorithm for orthogonalizing columns $q$ and $n+p$ can be obtained by replacing $p$ and $n + q$ with $q$ and $n + p$, respectively in function One-sided Concentric.

5.3 Numerical Experiments

The results of numerical experiments for computing a real symplectic SVD of a real symplectic matrix $A$ by the one-sided Jacobi algorithms are presented in this section. Here
\[
A \approx U \Sigma V^T
\]
is a real symplectic SVD of $A$ computed by our algorithms, where $U$ and $V$ are the computed real symplectic orthogonal matrices and $\Sigma$ is the computed real symplectic diagonal matrix.

We present results of two types of numerical experiments. In the first set of experiments, 50 random real symplectic matrices of dimension $2n \times 2n$ for $n = 30, 40, 50, 60, 70$, with condition number 10 were generated using an algorithm described by Jagger [21]. All six strategies: $\text{OS}_4, \text{OS}_{4s}, \text{OS}_5, \text{OS}_{5s}, \text{OS}_6, \text{OS}_{6s}$, were tested on each of these matrices and the results were compared. Only the average results are reported in Tables 5.2 - 5.7 and discussed below.

- **Sweeps** denotes the average number of sweeps needed by each strategy before the algorithm is terminated. There are a number of interesting trends exhibited in this row. First, as expected, the number of sweeps increases with increasing dimension for all strategies. Second, the number of sweeps for a strategy with the sorting angle is always less than that for a corresponding strategy with the small angle. Notice that strategies $\text{OS}_4$ and $\text{OS}_{4s}$ take more sweeps than other four strategies. However, we remind the reader that the total cost per full sweep incurred by strategies $\text{OS}_4$ and $\text{OS}_{4s}$ is approximately half that incurred by strategies $\text{OS}_5$, $\text{OS}_{5s}$, $\text{OS}_6$, $\text{OS}_{6s}$ as discussed in previous Chapter 4. In fact, the total number of iterations needed by strategies $\text{OS}_4$ and $\text{OS}_{4s}$ as shown in Table 5.1 is lower than that of the other four strategies. Here we present only the low, high, and average total number of iterations for test matrices of dimension $120 \times 120$ and $140 \times 140$. Moreover, from Table 5.1, if we compare the average total number of iterations for test matrices of dimension $140 \times 140$ among strategies $\text{OS}_4$, $\text{OS}_5$, and $\text{OS}_6$, then we have that the average total number of iterations of $\text{OS}_4$ is about 26.79% and 12.63% less than that of $\text{OS}_5$ and $\text{OS}_6$, respectively. Comparing the average total number of iterations for test matrices of dimension $140 \times 140$ among strategies $\text{OS}_{4s}$, $\text{OS}_{5s}$, and $\text{OS}_{6s}$, we have the average total number of iterations of $\text{OS}_{4s}$ is about 24.69% and 28.37% less than that of $\text{OS}_{5s}$ and $\text{OS}_{6s}$, respectively.

- The symplecticity of the generated matrices $A$ was tested via $\|AA^*-I\|_F$. It can be seen that this quantity was, on average, of order $10^{-14}$, so all strategies were tested on well structured matrices for all dimensions under consideration.

- $\|UU^*-I\|_F$ and $\|UU^T-I\|_F$ measure the average deviation of the computed left singular vector matrix from symplecticity and orthogonality, respectively. As we see, for all strategies and all dimensions, the loss of orthogonality in the computed left singular matrix $U$ was of order $10^{-13}$.

- $\|VV^*-I\|_F$ and $\|VV^T-I\|_F$ measure the average deviation of the computed
right singular vector matrix from symplecticity and orthogonality, respectively. The computed transformations $V$ were both real symplectic as well as orthogonal to within order $10^{-13}$.

- The row labeled $\text{diag.alg}$ reports the deviation of $\Sigma$ from being real symplectic diagonal matrix. Here $\text{diag.alg} = \max |x \cdot y - 1|$ where $x = [\sigma_1, \sigma_2, \ldots, \sigma_n]$, $y = [\sigma_{n+1}, \sigma_{n+2}, \ldots, \sigma_{2n}]$, and $\cdot$ denotes array multiplication (as in MATLAB). We see that each strategy provides real symplectic structure of $\Sigma$ very well for all dimensions. Note that all singular values of $A$ are real, and occur in pairs $(\sigma, 1/\sigma)$. To see how well MATLAB’s $\text{svd}$ function pairs the singular values, the pairing structure of singular values was tested. The results are shown in row labeled $\text{diag.Matlab}$. Let $v$ be the vector of the sorted singular values computed by $\text{svd}(A)$. Here $\text{diag.Matlab} = \max |x \cdot y - 1|$ where $x = v(1:n)$ and $y = v(2n:-1:n+1)$. Comparing these two rows, $\text{diag.alg}$ and $\text{diag.Matlab}$, we see from Tables 5.2 - 5.7 that the loss of pairing structure of singular values computed by our algorithms was of the order $10^{-14}$ while the loss of pairing structure of singular values computed by MATLAB was of the order $10^{-15}$ for all strategies and all dimensions under consideration.

- The row labeled $\text{rel.singular}$ records the maximum relative error as measured by

$$\text{rel.singular} = \max_j \frac{|\sigma_j^{\text{mat}} - \sigma_j^{\text{alg}}|}{|\sigma_j^{\text{mat}}|},$$

where $\{\sigma_j^{\text{mat}}\}$ and $\{\sigma_j^{\text{alg}}\}$ denote the singular values computed by MATLAB’s $\text{svd}$ function and by our strategies, respectively. As can be seen, the singular values computed by our strategies were comparable to MATLAB up to the order $10^{-14}$ for all dimensions under consideration.

<table>
<thead>
<tr>
<th>2n</th>
<th>120</th>
<th>140</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strategy</td>
<td>low</td>
<td>high</td>
</tr>
<tr>
<td>$\text{OS}_4$</td>
<td>$3.18 \times 10^4$</td>
<td>$3.28 \times 10^4$</td>
</tr>
<tr>
<td>$\text{OS}_{4s}$</td>
<td>$2.18 \times 10^4$</td>
<td>$2.24 \times 10^4$</td>
</tr>
<tr>
<td>$\text{OS}_5$</td>
<td>$3.94 \times 10^4$</td>
<td>$4.40 \times 10^4$</td>
</tr>
<tr>
<td>$\text{OS}_{5s}$</td>
<td>$2.76 \times 10^4$</td>
<td>$3.03 \times 10^4$</td>
</tr>
<tr>
<td>$\text{OS}_6$</td>
<td>$3.49 \times 10^4$</td>
<td>$3.72 \times 10^4$</td>
</tr>
<tr>
<td>$\text{OS}_{6s}$</td>
<td>$2.89 \times 10^4$</td>
<td>$3.19 \times 10^4$</td>
</tr>
</tbody>
</table>

Table 5.1: Statistical data for the total number of iterations

104

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|c|}
\hline
2n & 60 & 80 & 100 & 120 & 140 \\
\hline
sweeps & 10.30 & 10.80 & 11.15 & 11.40 & 11.80 \\
\hline
$\|A^*-I\|_F$ & $1.89 \times 10^{-14}$ & $2.38 \times 10^{-14}$ & $2.98 \times 10^{-14}$ & $3.47 \times 10^{-14}$ & $3.98 \times 10^{-14}$ \\
$\|U^*U^T-I\|_F$ & $3.77 \times 10^{-14}$ & $4.95 \times 10^{-14}$ & $6.44 \times 10^{-14}$ & $7.88 \times 10^{-14}$ & $9.21 \times 10^{-14}$ \\
$\|UU^T-I\|_F$ & $1.26 \times 10^{-13}$ & $2.30 \times 10^{-13}$ & $3.65 \times 10^{-13}$ & $5.22 \times 10^{-13}$ & $7.04 \times 10^{-13}$ \\
$\|VV^T-I\|_F$ & $7.84 \times 10^{-14}$ & $1.21 \times 10^{-13}$ & $1.80 \times 10^{-13}$ & $2.38 \times 10^{-13}$ & $3.14 \times 10^{-13}$ \\
$\|VV^*\|_F$ & $7.84 \times 10^{-14}$ & $1.21 \times 10^{-13}$ & $1.80 \times 10^{-13}$ & $2.38 \times 10^{-13}$ & $3.14 \times 10^{-13}$ \\
\hline
diag.alg & $1.46 \times 10^{-14}$ & $1.99 \times 10^{-14}$ & $2.53 \times 10^{-14}$ & $3.11 \times 10^{-14}$ & $3.75 \times 10^{-14}$ \\
\hline
diag.Matlab & $3.41 \times 10^{-15}$ & $4.18 \times 10^{-15}$ & $4.44 \times 10^{-15}$ & $4.86 \times 10^{-15}$ & $5.03 \times 10^{-15}$ \\
rel.singular & $8.17 \times 10^{-15}$ & $1.10 \times 10^{-14}$ & $1.40 \times 10^{-14}$ & $1.70 \times 10^{-14}$ & $2.00 \times 10^{-14}$ \\
\hline
\end{tabular}
\caption{One-sided Jacobi algorithm: $OS_4$}
\end{table}

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|c|}
\hline
2n & 60 & 80 & 100 & 120 & 140 \\
\hline
\hline
$\|A^*-I\|_F$ & $1.89 \times 10^{-14}$ & $2.38 \times 10^{-14}$ & $2.98 \times 10^{-14}$ & $3.47 \times 10^{-14}$ & $3.98 \times 10^{-14}$ \\
$\|U^*U^T-I\|_F$ & $3.31 \times 10^{-14}$ & $4.33 \times 10^{-14}$ & $5.49 \times 10^{-14}$ & $6.70 \times 10^{-14}$ & $7.76 \times 10^{-14}$ \\
$\|UU^T-I\|_F$ & $1.25 \times 10^{-13}$ & $2.16 \times 10^{-13}$ & $3.17 \times 10^{-13}$ & $4.82 \times 10^{-13}$ & $6.94 \times 10^{-13}$ \\
$\|VV^T-I\|_F$ & $7.51 \times 10^{-14}$ & $1.20 \times 10^{-13}$ & $1.73 \times 10^{-13}$ & $2.30 \times 10^{-13}$ & $2.94 \times 10^{-13}$ \\
$\|VV^*\|_F$ & $7.52 \times 10^{-14}$ & $1.20 \times 10^{-13}$ & $1.73 \times 10^{-13}$ & $2.30 \times 10^{-13}$ & $2.94 \times 10^{-13}$ \\
\hline
diag.alg & $1.38 \times 10^{-14}$ & $1.85 \times 10^{-14}$ & $2.33 \times 10^{-14}$ & $2.79 \times 10^{-14}$ & $3.30 \times 10^{-14}$ \\
\hline
diag.Matlab & $3.41 \times 10^{-15}$ & $4.18 \times 10^{-15}$ & $4.44 \times 10^{-15}$ & $4.86 \times 10^{-15}$ & $5.03 \times 10^{-15}$ \\
rel.singular & $8.06 \times 10^{-15}$ & $1.06 \times 10^{-14}$ & $1.29 \times 10^{-14}$ & $1.53 \times 10^{-14}$ & $1.76 \times 10^{-14}$ \\
\hline
\end{tabular}
\caption{One-sided Jacobi algorithm: $OS_4^s$}
\end{table}

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|c|}
\hline
2n & 60 & 80 & 100 & 120 & 140 \\
\hline
sweeps & 8.10 & 8.20 & 8.22 & 8.24 & 8.44 \\
\hline
$\|A^*-I\|_F$ & $1.77 \times 10^{-14}$ & $2.52 \times 10^{-14}$ & $2.88 \times 10^{-14}$ & $3.68 \times 10^{-14}$ & $4.21 \times 10^{-14}$ \\
$\|U^*U^T-I\|_F$ & $3.92 \times 10^{-14}$ & $5.62 \times 10^{-14}$ & $6.71 \times 10^{-14}$ & $8.41 \times 10^{-14}$ & $1.02 \times 10^{-13}$ \\
$\|UU^T-I\|_F$ & $1.43 \times 10^{-13}$ & $2.55 \times 10^{-13}$ & $4.16 \times 10^{-13}$ & $5.88 \times 10^{-13}$ & $8.09 \times 10^{-13}$ \\
$\|VV^T-I\|_F$ & $1.08 \times 10^{-13}$ & $1.76 \times 10^{-13}$ & $2.59 \times 10^{-13}$ & $3.42 \times 10^{-13}$ & $4.50 \times 10^{-13}$ \\
$\|VV^*\|_F$ & $1.08 \times 10^{-13}$ & $1.76 \times 10^{-13}$ & $2.59 \times 10^{-13}$ & $3.41 \times 10^{-13}$ & $4.50 \times 10^{-13}$ \\
\hline
diag.alg & $1.97 \times 10^{-14}$ & $2.81 \times 10^{-14}$ & $3.63 \times 10^{-14}$ & $4.36 \times 10^{-14}$ & $5.24 \times 10^{-14}$ \\
\hline
diag.Matlab & $2.00 \times 10^{-15}$ & $2.37 \times 10^{-15}$ & $2.64 \times 10^{-15}$ & $2.74 \times 10^{-15}$ & $3.10 \times 10^{-15}$ \\
rel.singular & $1.12 \times 10^{-14}$ & $1.47 \times 10^{-14}$ & $1.99 \times 10^{-14}$ & $2.28 \times 10^{-14}$ & $2.76 \times 10^{-14}$ \\
\hline
\end{tabular}
\caption{One-sided Jacobi algorithm: $OS_5$}
\end{table}

105

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
<table>
<thead>
<tr>
<th>2n</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>60</td>
<td>80</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>120</td>
<td>140</td>
<td></td>
</tr>
<tr>
<td>sweeps</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>6.08</td>
<td>6.12</td>
<td>6.24</td>
<td>6.44</td>
<td>6.50</td>
</tr>
<tr>
<td>$|AA^* - I|_F$</td>
<td>$1.77 \times 10^{-14}$</td>
<td>$2.52 \times 10^{-14}$</td>
<td>$2.88 \times 10^{-14}$</td>
<td>$3.68 \times 10^{-14}$</td>
<td>$4.21 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|UU^* - I|_F$</td>
<td>$3.43 \times 10^{-14}$</td>
<td>$4.83 \times 10^{-14}$</td>
<td>$5.63 \times 10^{-14}$</td>
<td>$7.18 \times 10^{-14}$</td>
<td>$8.55 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|UU^T - I|_F$</td>
<td>$1.66 \times 10^{-13}$</td>
<td>$2.84 \times 10^{-13}$</td>
<td>$4.66 \times 10^{-13}$</td>
<td>$6.64 \times 10^{-13}$</td>
<td>$9.08 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|VV^* - I|_F$</td>
<td>$9.46 \times 10^{-14}$</td>
<td>$1.40 \times 10^{-13}$</td>
<td>$2.02 \times 10^{-13}$</td>
<td>$2.80 \times 10^{-13}$</td>
<td>$3.56 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|VV^T - I|_F$</td>
<td>$9.46 \times 10^{-14}$</td>
<td>$1.40 \times 10^{-13}$</td>
<td>$2.02 \times 10^{-13}$</td>
<td>$2.80 \times 10^{-13}$</td>
<td>$3.56 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$1.68 \times 10^{-14}$</td>
<td>$2.11 \times 10^{-14}$</td>
<td>$2.68 \times 10^{-14}$</td>
<td>$3.19 \times 10^{-14}$</td>
<td>$3.83 \times 10^{-14}$</td>
</tr>
<tr>
<td>diag.Matlab</td>
<td>$2.00 \times 10^{-15}$</td>
<td>$2.37 \times 10^{-15}$</td>
<td>$2.64 \times 10^{-15}$</td>
<td>$2.74 \times 10^{-15}$</td>
<td>$3.10 \times 10^{-15}$</td>
</tr>
<tr>
<td>rel.singular</td>
<td>$9.38 \times 10^{-15}$</td>
<td>$1.14 \times 10^{-14}$</td>
<td>$1.44 \times 10^{-14}$</td>
<td>$1.73 \times 10^{-14}$</td>
<td>$2.07 \times 10^{-14}$</td>
</tr>
</tbody>
</table>

Table 5.5: One-sided Jacobi algorithm: $\mathcal{O}S_{5s}$

<table>
<thead>
<tr>
<th>2n</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>60</td>
<td>80</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>120</td>
<td>140</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sweeps</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>6.74</td>
<td>6.98</td>
<td>7.06</td>
<td>7.08</td>
<td>7.60</td>
</tr>
<tr>
<td>$|AA^* - I|_F$</td>
<td>$1.77 \times 10^{-14}$</td>
<td>$2.52 \times 10^{-14}$</td>
<td>$2.88 \times 10^{-14}$</td>
<td>$3.68 \times 10^{-14}$</td>
<td>$4.21 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|UU^* - I|_F$</td>
<td>$3.63 \times 10^{-14}$</td>
<td>$5.36 \times 10^{-14}$</td>
<td>$6.53 \times 10^{-14}$</td>
<td>$7.98 \times 10^{-14}$</td>
<td>$9.47 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|UU^T - I|_F$</td>
<td>$1.16 \times 10^{-13}$</td>
<td>$2.49 \times 10^{-13}$</td>
<td>$3.79 \times 10^{-13}$</td>
<td>$5.35 \times 10^{-13}$</td>
<td>$7.79 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|VV^* - I|_F$</td>
<td>$8.83 \times 10^{-14}$</td>
<td>$1.54 \times 10^{-13}$</td>
<td>$2.12 \times 10^{-13}$</td>
<td>$3.00 \times 10^{-13}$</td>
<td>$3.88 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|VV^T - I|_F$</td>
<td>$8.82 \times 10^{-14}$</td>
<td>$1.54 \times 10^{-13}$</td>
<td>$2.12 \times 10^{-13}$</td>
<td>$3.00 \times 10^{-13}$</td>
<td>$3.88 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$1.72 \times 10^{-14}$</td>
<td>$2.37 \times 10^{-14}$</td>
<td>$2.94 \times 10^{-14}$</td>
<td>$3.72 \times 10^{-14}$</td>
<td>$4.45 \times 10^{-14}$</td>
</tr>
<tr>
<td>diag.Matlab</td>
<td>$2.00 \times 10^{-15}$</td>
<td>$2.37 \times 10^{-15}$</td>
<td>$2.64 \times 10^{-15}$</td>
<td>$2.74 \times 10^{-15}$</td>
<td>$3.10 \times 10^{-15}$</td>
</tr>
<tr>
<td>rel.singular</td>
<td>$9.38 \times 10^{-15}$</td>
<td>$1.29 \times 10^{-14}$</td>
<td>$1.61 \times 10^{-14}$</td>
<td>$1.97 \times 10^{-14}$</td>
<td>$2.34 \times 10^{-14}$</td>
</tr>
</tbody>
</table>

Table 5.6: One-sided Jacobi algorithm: $\mathcal{O}S_{6s}$

<table>
<thead>
<tr>
<th>2n</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>60</td>
<td>80</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>120</td>
<td>140</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sweeps</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>6.10</td>
<td>6.25</td>
<td>6.40</td>
<td>6.74</td>
<td>6.94</td>
</tr>
<tr>
<td>$|AA^* - I|_F$</td>
<td>$1.77 \times 10^{-14}$</td>
<td>$2.52 \times 10^{-14}$</td>
<td>$2.88 \times 10^{-14}$</td>
<td>$3.68 \times 10^{-14}$</td>
<td>$4.21 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|UU^* - I|_F$</td>
<td>$3.53 \times 10^{-14}$</td>
<td>$4.94 \times 10^{-14}$</td>
<td>$5.97 \times 10^{-14}$</td>
<td>$7.46 \times 10^{-14}$</td>
<td>$8.84 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|UU^T - I|_F$</td>
<td>$1.29 \times 10^{-13}$</td>
<td>$2.44 \times 10^{-13}$</td>
<td>$3.65 \times 10^{-13}$</td>
<td>$5.26 \times 10^{-13}$</td>
<td>$7.51 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|VV^* - I|_F$</td>
<td>$9.58 \times 10^{-14}$</td>
<td>$1.45 \times 10^{-13}$</td>
<td>$2.19 \times 10^{-13}$</td>
<td>$2.88 \times 10^{-13}$</td>
<td>$3.99 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|VV^T - I|_F$</td>
<td>$9.58 \times 10^{-14}$</td>
<td>$1.45 \times 10^{-13}$</td>
<td>$2.19 \times 10^{-13}$</td>
<td>$2.88 \times 10^{-13}$</td>
<td>$3.99 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$1.66 \times 10^{-14}$</td>
<td>$2.16 \times 10^{-14}$</td>
<td>$2.86 \times 10^{-14}$</td>
<td>$3.34 \times 10^{-14}$</td>
<td>$4.25 \times 10^{-14}$</td>
</tr>
<tr>
<td>diag.Matlab</td>
<td>$2.00 \times 10^{-15}$</td>
<td>$2.37 \times 10^{-15}$</td>
<td>$2.64 \times 10^{-15}$</td>
<td>$2.74 \times 10^{-15}$</td>
<td>$3.10 \times 10^{-15}$</td>
</tr>
<tr>
<td>rel.singular</td>
<td>$1.00 \times 10^{-14}$</td>
<td>$1.16 \times 10^{-14}$</td>
<td>$1.54 \times 10^{-14}$</td>
<td>$1.81 \times 10^{-14}$</td>
<td>$2.28 \times 10^{-14}$</td>
</tr>
</tbody>
</table>

Table 5.7: One-sided Jacobi algorithm: $\mathcal{O}S_{6s}$
A second set of experiments was designed to see how accurately our strategies computed the singular values of a real symplectic test matrix with pre-chosen singular values. Then we compared the known singular values to singular values computed by our strategies and to those computed by MATLAB’s `svd` function. Some of the experiments are shown below.

**Example 9.** A real symplectic matrix $A$ was generated with singular values $10, 5, 2, \frac{1}{2}, \frac{1}{5}, \frac{1}{10}$. In this case $A$ is a well-conditioned matrix with condition number 100. Table 5.8 shows the exact singular values and the relative errors in the singular value approximations computed by our algorithms and by MATLAB’s `svd` function.

![Table 5.8: Accuracy of the singular values 10, 5, 2, \frac{1}{2}, \frac{1}{5}, \frac{1}{10}](image)

**Example 10.** A real symplectic matrix $A$ was generated with singular values $a, b, c, \frac{1}{a}, \frac{1}{b}, \frac{1}{c}$ where $[a, b, c] = [3, 3, 3] + [9.2, 8.7, 5.4] \times 10^{-15}$. Here $A$ is a well-conditioned matrix and has two clusters of singular values of “multiplicity” 3—singular values near 3 and near $\frac{1}{3}$. The exact singular values and the relative errors are given in Table 5.9.

![Table 5.9: Accuracy of the singular values near 3 and $\frac{1}{3}$](image)

**Example 11.** A real symplectic matrix $A$ was generated with singular values: $\sigma_j = a, b, c, \frac{1}{a}, \frac{1}{b}, \frac{1}{c}$ where $[a, b, c] = [100, 100, 100] + [7.3, 3.2, 2.6] \times 10^{-15}$. In this case, $A$ is an ill-conditioned matrix with condition number $10^4$, and has two cluster
of singular values, one around 100, of "multiplicity" 3, and another around \( \frac{1}{100} \) of the same "multiplicity". The exact singular values and the relative errors are reported in Table 5.10.

<table>
<thead>
<tr>
<th>( \sigma_j )</th>
<th>( \mathbf{OS}_4 )</th>
<th>( \mathbf{OS}_{4a} )</th>
<th>( \mathbf{OS}_5 )</th>
<th>( \mathbf{OS}_{5a} )</th>
<th>( \mathbf{OS}_6 )</th>
<th>( \mathbf{OS}_{6a} )</th>
<th>Matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/( a )</td>
<td>1.3e-13</td>
<td>2.8e-13</td>
<td>2.4e-13</td>
<td>2.7e-13</td>
<td>3.5e-13</td>
<td>2.2e-13</td>
<td>5.6e-13</td>
</tr>
<tr>
<td>1/( b )</td>
<td>2.4e-15</td>
<td>2.0e-13</td>
<td>7.4e-14</td>
<td>2.5e-14</td>
<td>1.8e-13</td>
<td>1.8e-13</td>
<td>7.8e-13</td>
</tr>
<tr>
<td>1/( c )</td>
<td>3.0e-13</td>
<td>2.8e-14</td>
<td>2.1e-13</td>
<td>2.3e-13</td>
<td>2.4e-13</td>
<td>4.6e-14</td>
<td>1.0e-12</td>
</tr>
<tr>
<td>( c )</td>
<td>0</td>
<td>0</td>
<td>2.8e-16</td>
<td>2.8e-16</td>
<td>4.2e-16</td>
<td>1.4e-16</td>
<td>4.2e-16</td>
</tr>
<tr>
<td>1/( b )</td>
<td>2.8e-16</td>
<td>1.4e-16</td>
<td>0</td>
<td>2.8e-16</td>
<td>0</td>
<td>4.2e-16</td>
<td></td>
</tr>
<tr>
<td>1/( a )</td>
<td>4.3e-16</td>
<td>1.4e-16</td>
<td>1.4e-16</td>
<td>4.2e-16</td>
<td>2.8e-16</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.10: Accuracy of the singular values near 100 and \( \frac{1}{100} \)

**Example 12.** A real symplectic matrix \( \mathbf{A} \) was generated with singular values \( 10^4, 10^2, 10, \frac{1}{10}, \frac{1}{10^2}, \frac{1}{10^4} \). In this case, \( \mathbf{A} \) is an ill-conditioned matrix with condition number \( 10^8 \). The exact singular values and the relative errors are reported in Table 5.11.

<table>
<thead>
<tr>
<th>( \sigma_j )</th>
<th>( \mathbf{OS}_4 )</th>
<th>( \mathbf{OS}_{4a} )</th>
<th>( \mathbf{OS}_5 )</th>
<th>( \mathbf{OS}_{5a} )</th>
<th>( \mathbf{OS}_6 )</th>
<th>( \mathbf{OS}_{6a} )</th>
<th>Matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/( 10^4 )</td>
<td>6.5e-10</td>
<td>6.5e-10</td>
<td>1.1e-9</td>
<td>1.7e-10</td>
<td>1.0e-9</td>
<td>1.7e-10</td>
<td>4.3e-10</td>
</tr>
<tr>
<td>1/( 10^2 )</td>
<td>6.5e-13</td>
<td>2.4e-15</td>
<td>1.2e-11</td>
<td>8.0e-12</td>
<td>1.8e-11</td>
<td>8.6e-12</td>
<td>1.0e-11</td>
</tr>
<tr>
<td>1/10</td>
<td>3.0e-12</td>
<td>3.0e-12</td>
<td>6.3e-13</td>
<td>6.6e-13</td>
<td>1.0e-12</td>
<td>5.8e-13</td>
<td>4.2e-12</td>
</tr>
<tr>
<td>10</td>
<td>1.6e-15</td>
<td>2.3e-15</td>
<td>4.6e-15</td>
<td>3.5e-16</td>
<td>3.9e-15</td>
<td>1.7e-14</td>
<td>2.4e-15</td>
</tr>
<tr>
<td>( 10^2 )</td>
<td>1.4e-15</td>
<td>1.3e-15</td>
<td>8.5e-16</td>
<td>2.8e-16</td>
<td>1.5e-16</td>
<td>4.2e-16</td>
<td>7.1e-16</td>
</tr>
<tr>
<td>( 10^4 )</td>
<td>3.6e-16</td>
<td>0</td>
<td>7.2e-16</td>
<td>1.8e-16</td>
<td>7.2e-16</td>
<td>5.4e-16</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 5.11: Accuracy of the singular values \( 10^4, 10^2, 10, \frac{1}{10}, \frac{1}{10^2}, \frac{1}{10^4} \)

**Example 13.** A real \( 20 \times 20 \) symplectic matrix \( \mathbf{A} \) was generated with singular values \( 2, \frac{1}{2}, 4, \frac{1}{4}, 6, \frac{1}{6}, 7, \frac{1}{7}, a, b, c, \frac{1}{a}, \frac{1}{b}, \frac{1}{c}, x, y, z, \frac{1}{x}, \frac{1}{y}, \frac{1}{z} \) where \( [a, b, c] = [3, 3, 3] + [0.9, 4.9, 8.4] \times 10^{-15} \) and \( [x, y, z] = [5, 5, 5] + [4.4, 5.3, 6.2] \times 10^{-15} \). In this case, \( \mathbf{A} \) is well-conditioned and has 4 clusters of singular values of "multiplicity" 3 — singular values near 3, \( \frac{1}{3} \), 5, and \( \frac{1}{5} \). Table 5.12 shows the exact singular values and the relative errors in the singular value computed by our algorithms and by MATLAB's \texttt{svd} function in each set of clusters.

The results from all examples as shown in Tables 5.8 - 5.12 report that the singular values computed by our strategies and by MATLAB are comparable. In facts, when the generated matrix \( \mathbf{A} \) is well-conditioned as in Examples 9, 10, and 13, our strategies give the singular values with the relative errors as same as MATLAB's \texttt{svd} function to within \( 10^{-15} \). Also, when the generated matrix \( \mathbf{A} \) is ill-conditioned.
as given in Examples 11 and 12, our strategies compute the large singular values with as high relative accuracy as by MATLAB’s \texttt{svd} function. However, the loss of accuracy of the small singular values computed by our algorithms is the same as MATLAB’s \texttt{svd} function when \( A \) is ill-conditioned. Moreover, when \( A \) has the clusters of singular values as in Examples 10, 11, and 13, our algorithms yield singular values with the same relative error as does by MATLAB’s \texttt{svd} function for each cluster.

<table>
<thead>
<tr>
<th>( \sigma_i )</th>
<th>\texttt{OS}_4</th>
<th>\texttt{OS}_{4\times}</th>
<th>\texttt{OS}_5</th>
<th>\texttt{OS}_{5\times}</th>
<th>\texttt{OS}_6</th>
<th>\texttt{OS}_{6\times}</th>
<th>\texttt{Matlab}</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a )</td>
<td>5.9e-16</td>
<td>1.5e-15</td>
<td>4.4e-15</td>
<td>2.6e-15</td>
<td>4.4e-15</td>
<td>4.0e-15</td>
<td>1.3e-15</td>
</tr>
<tr>
<td>( b )</td>
<td>4.4e-16</td>
<td>3.0e-16</td>
<td>9.3e-15</td>
<td>2.6e-15</td>
<td>5.3e-15</td>
<td>4.0e-15</td>
<td>0</td>
</tr>
<tr>
<td>( c )</td>
<td>8.9e-16</td>
<td>8.9e-16</td>
<td>6.2e-15</td>
<td>0</td>
<td>4.8e-15</td>
<td>3.1e-15</td>
<td>1.7e-15</td>
</tr>
<tr>
<td>( 1/a )</td>
<td>3.3e-16</td>
<td>0</td>
<td>1.1e-15</td>
<td>3.3e-15</td>
<td>7.7e-15</td>
<td>6.1e-15</td>
<td>2.7e-15</td>
</tr>
<tr>
<td>( 1/b )</td>
<td>9.9e-16</td>
<td>3.3e-16</td>
<td>1.1e-15</td>
<td>1.6e-15</td>
<td>4.4e-15</td>
<td>3.3e-15</td>
<td>1.1e-15</td>
</tr>
<tr>
<td>( 1/c )</td>
<td>8.3e-16</td>
<td>6.7e-16</td>
<td>1.1e-15</td>
<td>1.1e-15</td>
<td>7.7e-15</td>
<td>7.2e-15</td>
<td>0</td>
</tr>
<tr>
<td>( x )</td>
<td>3.6e-16</td>
<td>1.1e-15</td>
<td>0</td>
<td>8.8e-16</td>
<td>6.2e-15</td>
<td>6.2e-15</td>
<td>8.8e-16</td>
</tr>
<tr>
<td>( y )</td>
<td>3.6e-16</td>
<td>7.1e-16</td>
<td>8.8e-15</td>
<td>1.7e-15</td>
<td>6.2e-15</td>
<td>8.8e-15</td>
<td>1.7e-15</td>
</tr>
<tr>
<td>( z )</td>
<td>3.6e-16</td>
<td>8.9e-16</td>
<td>8.8e-15</td>
<td>3.5e-15</td>
<td>5.3e-15</td>
<td>9.7e-15</td>
<td>0</td>
</tr>
<tr>
<td>( 1/x )</td>
<td>1.4e-16</td>
<td>1.7e-15</td>
<td>3.6e-16</td>
<td>9.9e-16</td>
<td>5.5e-16</td>
<td>4.4e-16</td>
<td>4.1e-16</td>
</tr>
<tr>
<td>( 1/y )</td>
<td>8.3e-16</td>
<td>1.3e-15</td>
<td>3.6e-16</td>
<td>9.9e-16</td>
<td>4.1e-16</td>
<td>4.4e-16</td>
<td>3.8e-16</td>
</tr>
<tr>
<td>( 1/z )</td>
<td>8.3e-16</td>
<td>8.3e-16</td>
<td>3.3e-16</td>
<td>9.9e-16</td>
<td>2.7e-16</td>
<td>2.7e-16</td>
<td>3.6e-16</td>
</tr>
</tbody>
</table>

Table 5.12: Accuracy of the singular values near 3, \( \frac{1}{3} \), 5 and \( \frac{1}{5} \)
Chapter 6
Computing the Complex Symplectic SVD

Our attention now turns to finding algorithms for computing a complex symplectic SVD of a given complex symplectic matrix. Using the fact that such a matrix is the complex generalization of a real symplectic matrix, the algorithms that we developed for a real symplectic SVD can be extended to algorithms for computing a complex symplectic SVD. In Chapters 4 and 5 we presented two ways for finding a real symplectic SVD — a two-sided Jacobi method using the polar decomposition and a direct one-sided Jacobi method. We show how to adapt both these methods to the complex case.

6.1 Using the Polar Decomposition

Let the \( 2n \times 2n \) complex symplectic matrix \( A \) have the polar decomposition \( A = UH \), where \( U \) is the unitary factor and \( H \) is the Hermitian positive definite factor. As remarked earlier, both \( U \) and \( H \) are also complex symplectic, and the problem of finding structure preserving algorithms for computing a complex symplectic SVD of \( A \) can be reduced to finding structure preserving algorithms for computing a structured spectral decomposition of \( H \). Once again, one can construct the complex symplectic SVD of \( A \) as in (1.16), which, for convenience, we recall here:

\[
A = UH, \quad \text{where} \quad U^*U = I, \quad H^* = H \\
= U(Q\Sigma Q^*), \quad \text{substituting} \quad H = Q\Sigma Q^* \quad \text{with} \quad Q^*Q = I \\
= (UQ)\Sigma Q^*, \\
= W\Sigma Q^* \quad \text{where} \quad W = UQ. \quad (6.1)
\]

As discussed in the real case, the complex symplectic SVD of \( A \) can be achieved in two stages: the first stage computes the complex symplectic unitary \( U \), and
the structured spectral decomposition of the complex symplectic Hermitian $H$ is computed in the second stage. As mentioned earlier, structure preserving iterations for computing the complex symplectic unitary polar factor $U$ were given by Higham, Mackey, Mackey, and Tisseur in [16]. Here we use the Newton iteration:

$$X_{k+1} = \frac{1}{2}(X_k + X_k^{-*}), \quad X_0 = A,$$

(6.2)

to compute the polar factor $U$. The complex symplectic Hermitian factor can then be determined as $H = U^*A$. In order to compute a complex symplectic SVD of $A$, structure preserving algorithms for computing a structured spectral decomposition of the doubly structured factor $H$ are needed. The following section provides such algorithms.

### 6.1.1 Structured Spectral Decomposition of Complex Symplectic Hermitian Matrices

As we did in the real case, the algorithms we develop are Jacobi-like. The basic step of these methods is to diagonalize $2 \times 2$ subproblems in a structure preserving way. The position of an off-diagonal pivot element chosen from the $2n \times 2n$ complex symplectic Hermitian matrix determines the location of a subproblem. We begin by describing the appropriate tools and the appropriate sweep designs.

Recall that for the real case, the sweep designs of strategies $S_4$, $S_5$ and $S_6$ were the best among those described in Section 4.2. We therefore consider only these three designs for the complex case. Thus the only task that remains unfinished is that of choosing appropriate tools for our algorithms.

Preserving the complex symplectic Hermitian structure of a matrix limits our tools to transformations in the complex symplectic unitary group $SpU(2n, \mathbb{C})$ defined in (1.14). We start by designing a $2 \times 2$ complex symplectic unitary matrix that diagonalizes a $2 \times 2$ Hermitian matrix.

In [26], an explicit characterization of the matrices in $SpU(2, \mathbb{C})$ was given. That is, a $2 \times 2$ complex symplectic unitary matrix is of the form

$$\begin{bmatrix}
\alpha & -\beta \\
-\beta & \alpha
\end{bmatrix}, \quad \alpha, \beta \in \mathbb{C}, \quad |\alpha|^2 + |\beta|^2 = 1.$$

Let $R$ be a matrix of the form

$$R = \begin{bmatrix}
e^{i\alpha} \cos \theta & e^{i\beta} \sin \theta \\
-e^{-i\beta} \sin \theta & e^{-i\alpha} \cos \theta
\end{bmatrix}, \quad \alpha, \beta, \theta \in \mathbb{R}.$$

(6.3)

Then clearly $R \in SpU(2, \mathbb{C})$. Observe that when $\alpha = \beta = 0$, $R$ reduces to a real plane rotation. We will call matrices of the form (6.3) with $\alpha = \beta$ as complex symplectic rotations. The following proposition characterizes all matrices of the form as in (6.3) that diagonalize a given $2 \times 2$ Hermitian matrix.
Proposition 6.1. Let $A$ be a $2 \times 2$ Hermitian matrix of the form

$$A = \begin{bmatrix} a & z \\ \bar{z} & b \end{bmatrix},$$

where $z = re^{i\phi} \neq 0$, and $a, b, r, \phi \in \mathbb{R}$,

and let $R$ be a $2 \times 2$ complex symplectic unitary matrix of the form

$$R = \begin{bmatrix} e^{i\alpha} \cos \theta & e^{i\beta} \sin \theta \\ -e^{-i\beta} \sin \theta & e^{-i\alpha} \cos \theta \end{bmatrix}, \quad \alpha, \beta, \theta \in \mathbb{R}.$$

Then $R^*AR$ is diagonal if $\alpha + \beta \equiv \phi (\text{mod } \pi)$ and either

(i) $\cos 2\theta = 0$, when $a = b$,

or (ii) $\tan 2\theta = \pm \frac{2r}{b - a}$, when $a \neq b$.

Proof. Setting the off-diagonal of $\hat{A} = R^*AR$ to zero gives

$$e^{i(\beta - \alpha)}(a - b) \cos \theta \sin \theta - \bar{z}e^{i2\beta} \sin^2 \theta + ze^{-i2\alpha} \cos^2 \theta = 0. \quad (6.4)$$

Multiplying (6.4) by $e^{i(\alpha - \beta)}$ yields

$$(a - b) \cos \theta \sin \theta - \bar{z}e^{i(\alpha + \beta)} \sin^2 \theta + ze^{-i(\alpha + \beta)} \cos^2 \theta = 0. \quad (6.5)$$

Let $p = ze^{-i(\alpha + \beta)}$. Equation (6.5) can be rewritten as

$$(a - b) \cos \theta \sin \theta - \bar{p} \sin^2 \theta + p \cos^2 \theta = 0. \quad (6.6)$$

Using the identities $\sin^2 \theta = \frac{1}{2}(1 - \cos 2\theta)$ and $\cos^2 \theta = \frac{1}{2}(1 + \cos 2\theta)$ in (6.6), one obtains

$$(a - b) \cos \theta \sin \theta - \frac{\bar{p}}{2}(1 - \cos 2\theta) + \frac{p}{2}(1 + \cos 2\theta) = 0,$$

$$(a - b) \cos \theta \sin \theta + \frac{1}{2}(p + \bar{p}) \cos 2\theta + \frac{1}{2}(p - \bar{p}) = 0,$$

$$(a - b) \cos \theta \sin \theta + \text{Re}(p) \cos 2\theta + i\text{Im}(p) = 0,$$

where $\text{Re}(p)$ and $\text{Im}(p)$ denote the real and imaginary part of $p$, respectively. Since $p = ze^{-i(\alpha + \beta)}$ and $z = re^{i\phi}$, we have

$$(a - b) \cos \theta \sin \theta + \text{Re}(re^{-i(\alpha + \beta - \phi)}) \cos 2\theta + i\text{Im}(re^{-i(\alpha + \beta - \phi)}) = 0,$$

$$(a - b) \cos \theta \sin \theta + r \cos(\alpha + \beta - \phi) \cos 2\theta - ir \sin(\alpha + \beta - \phi) = 0. \quad (6.7)$$

Equating real and imaginary parts to zero, we see that

$$(a - b) \cos \theta \sin \theta + r \cos(\alpha + \beta - \phi) \cos 2\theta = 0, \quad (6.8)$$
and
\[ r \sin(\alpha + \beta - \phi) = 0. \]  
(6.9)

Since \( z \neq 0 \) by assumption, i.e. \( r \neq 0 \), from (6.9) we get
\[ \sin(\alpha + \beta - \phi) = 0. \]

That is, \( \alpha + \beta - \phi = k\pi, k = 0, \pm 1, \pm 2, \ldots \), and hence
\[ \alpha + \beta \equiv \phi \mod(\pi). \]  
(6.10)

Using the identity \( \sin 2\theta = 2\sin \theta \cos \theta \) in (6.8) gives
\[ r \cos(\alpha + \beta - \phi) \cos 2\theta = \frac{1}{2}(b - a) \sin 2\theta. \]  
(6.11)

But \( \alpha + \beta \equiv \phi \mod(\pi) \), so \( \cos(\alpha + \beta - \phi) = \pm 1 \), and hence (6.11) becomes
\[ \pm r \cos 2\theta = \frac{1}{2}(b - a) \sin 2\theta. \]  
(6.12)

If \( a = b \), then from (6.12) one obtains \( \cos 2\theta = 0 \). On the other hand, if \( a \neq b \), then we have \( \tan 2\theta = \frac{\pm 2r}{b - a} \).

We now describe \( 2n \times 2n \) complex symplectic unitary transformations used in our algorithms. They are built using the \( 2 \times 2 \) complex symplectic rotation \( R \) given in (6.3) with \( \alpha = \beta \), and were described by Mackey, Mackey, and Tisseur in [26]. For convenience, we use the abbreviations \( c = e^{i\alpha} \cos \theta, s = e^{i\alpha} \sin \theta \), so \( R = [c \ s; -\bar{s} \ c] \).

Embedding \( R \) as a principal submatrix into rows and columns \( p, n + p \) of a \( 2n \times 2n \) identity matrix yields a \( 2n \times 2n \) complex symplectic plane rotation

\[
Q = \begin{bmatrix}
p & \bar{s} \\
p & \bar{s}
\end{bmatrix} \quad \in \text{SpU}(2n, \mathbb{C}).
\]  
(6.13)

The effect of a similarity transformation by \( Q \) on a given Hermitian matrix \( A \) is analogous to that by a real symplectic plane rotation on a symmetric matrix as described in Section 2.1. In particular, zeroing the off-diagonal entries \( (p, n + p) \)
and \((n + p, p)\) of \(\hat{A} = Q^*AQ\) moves all weight of these pivot elements onto the diagonal entries \((p, p)\) and \((n + p, n + p)\); no weight leaks out from any diagonal element into the off-diagonal part.

Embedding \(R\) into rows and columns \(p, q,\) and \(\bar{R},\) the complex conjugate of \(R,\) into rows and columns \(n + p, n + q\) of \(I_{2n}\) yields

\[
Q = \begin{pmatrix}
  p & q & n+p & n+q \\
  p & & c & -\bar{s} \\
  q & c & & -\bar{c} \\
  n+p & & & \\
  n+q & & & \\
\end{pmatrix} \in SpU(2n, \mathbb{C}). \tag{6.14}
\]

We will refer to this matrix as a \(2n \times 2n\) complex symplectic double rotation or a double rotation for short. The effect of a similarity transformation by \(Q\) on a given Hermitian matrix \(A\) is analogous to that by a real symplectic double rotation with direct sum embedding on a symmetric matrix as described in Section 2.3.1. For the case when the aim of the similarity is to set \(\hat{a}_{pq}\) and \(\hat{a}_{qp}\) to zero, the rotation in the upper diagonal block is the active one that makes these pivots zero; all the weight of the pivots is moved to the diagonal entries \((p, p)\) and \((q, q)\) of \(\hat{A}\). However, there is liable to be some leakage of weight from the diagonal entries \((n + p, n + p)\) and \((n + q, n + q)\) into the off-diagonal entries \((n + p, n + q)\) and \((n + q, n + p)\) by the action of the inactive rotation in the lower diagonal block.

Embedding one copy of \(R\) into rows and columns \(p, n + q,\) and a second copy of \(R\) (not \(\bar{R}\)) in rows and columns \(q, n + p\) yields

\[
Q = \begin{pmatrix}
  p & q & n+p & n+q \\
  p & & c & \bar{s} \\
  q & c & & s \\
  n+p & & & \\
  n+q & & & \\
\end{pmatrix} \in SpU(2n, \mathbb{C}). \tag{6.15}
\]

We will refer to this matrix as a \(2n \times 2n\) complex symplectic concentric rotation or a concentric rotation for short. The effect of a similarity transformation by \(Q\) on a given Hermitian matrix \(A\) is analogous to that by a real symplectic double rotation with concentric embedding on a symmetric matrix as described in Section
2.3.2. That is, if the aim of similarity is to set the off-diagonal elements \( a_{p,n+q} \) and \( a_{n+q,p} \) of \( A \) to zero, then the outer rotation is the active rotation that introduces zeroes in positions \((p,n + q)\) and \((n + q, p)\). All the weight of the pivots moves into the diagonal entries \((p,p)\) and \((n + q,n + q)\). Simultaneously, there is liable to be some leakage of weight from the diagonal entries \((g,q)\) and \((n + p, n + p)\) into the off-diagonal entries \((g,n + p)\) and \((n + p, q)\) by the action of the inner, inactive rotation.

At this point we have all the tools needed for each chosen sweep design, and we can proceed to build structure preserving algorithms for computing a structured spectral decomposition of complex symplectic Hermitian matrices.

Now comes a crucial observation. In practice, the computed complex symplectic Hermitian factor \( H \) from the polar decomposition of a complex symplectic matrix \( A \) is neither exactly complex symplectic nor exactly Hermitian. Rather, \( H \) is close to being complex symplectic and close to being Hermitian: numerical experiments show that \( \| H H^* - I \|_F \approx 10^{-14} \) and \( \| H - H^* \|_F \approx 10^{-14} \). The fact that \( H \) is not exactly complex symplectic does not directly cause us any numerical difficulty. But the lack of Hermitianness has to be addressed. We first focus on the non-reality of the diagonal entries.

Consider a nearly \( 2 \times 2 \) Hermitian matrix of the form

\[
A = \begin{bmatrix} a + i\delta_a & z \\ \bar{z} & b + i\delta_b \end{bmatrix}, \quad a, b, \delta_a, \delta_b \in \mathbb{R}, \quad |\delta_a| \ll \text{Re}|a|, |\delta_b| \ll \text{Re}|b|, \\
z = re^{i\phi} \neq 0, \quad r, \phi \in \mathbb{R}.
\]

One might ask: can \( R \) given in (6.3) be used to diagonalize the nearly Hermitian matrix \( A \)? To answer this question we consider the following. By replacing \( a \) and \( b \) in (6.7) with \( a + i\delta_a \) and \( b + i\delta_b \), respectively, and setting \( \alpha = \beta \) we obtain

\[
\{(a - b) \cos \theta \sin \theta + r \cos(2\alpha - \phi) \cos 2\theta \} - i \{(\delta_b - \delta_a) + r \sin(2\alpha - \phi)\} = 0.
\]

This implies that \( \alpha \) and \( \theta \) must satisfy

\[
(a - b) \cos \theta \sin \theta + r \cos(2\alpha - \phi) \cos 2\theta = 0
\]

and

\[
(\delta_b - \delta_a) + r \sin(2\alpha - \phi) = 0 \quad (6.16)
\]

Since \( r \neq 0 \) by assumption, from (6.16) we have

\[
\sin(2\alpha - \phi) = \frac{\delta_a - \delta_b}{r}. \quad (6.17)
\]

If \( \frac{\delta_a - \delta_b}{r} \notin [-1,1] \), then there is no solution of (6.17), and hence no such transformation \( R \) exists. Let us give an example for such a possibility. Consider a case when \( \delta_a - \delta_b \approx 2\epsilon s \), and \( z = \rho_1 + i\rho_2 \), where \( \rho_1, \rho_2 \approx \epsilon s \). Then

\[
\frac{\delta_a - \delta_b}{r} = \frac{\delta_a - \delta_b}{\sqrt{\rho_1^2 + \rho_2^2}} \approx \frac{2\epsilon s}{\sqrt{2}\epsilon s} \approx \sqrt{2} \notin [-1,1].
\]

115

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Since the problem of having a nearly $2n \times 2n$ Hermitian matrix $H$ with non-real diagonal entries as input to our algorithms cannot be avoided, we need to replace $H$ by a nearby matrix and then apply our algorithm to that matrix. We consider the 3 choices below:

- $\tilde{H}_1 = \frac{1}{2}(H + H^*)$
- $\tilde{H}_2 = [\tilde{h}_{ij}]$ with $\tilde{h}_{ij} = h_{ij}$ for $i \neq j$, and $\tilde{h}_{ii} = (\text{sign}\, \text{Re}(h_{ii}))|h_{ii}|$ for $i = 1, 2, \ldots, 2n$.
- $\tilde{H}_3 = [h_{ij}]$ with $\tilde{h}_{ij} = h_{ij}$ for $i \neq j$, and $\tilde{h}_{ii} = \text{Re}(h_{ii})$ for $i = 1, 2, \ldots, 2n$.

Using MATLAB programs written by N. Mackey that encode a quaternion based algorithm given in [24] we generated 100 random complex symplectic matrices $A$ of dimension $2n \times 2n$ for $n = 30, 40, 50, 60, 70$. After computing the polar factor $U$ of each of these matrices by Newton iteration as given in (6.2), the positive definite Hermitian factor $H$ was determined by $H = U^*A$. Due to roundoff error, the diagonal entries of $H$ have small imaginary part of the order of $10^{-16}$ on average. For each of these matrices $H$, the corresponding nearby matrices $\tilde{H}_1, \tilde{H}_2, \tilde{H}_3$ were computed. Note that since $H$ is positive definite, $\text{sign}\, \text{Re}(|h_{ii}|)$ should be positive. The symplecticity of $\tilde{H}_1$ is reported in Table 6.1 below. We see that the symplecticity of $\tilde{H}_2$ is the same as that of $\tilde{H}_3$ for all dimensions under consideration. Also, the symplectic structure of $\tilde{H}_2$ and $\tilde{H}_3$ is on average slightly better than that of $\tilde{H}_1$. We therefore choose the third option of replacing diagonal entries by their real parts.

| 2n  | $||\tilde{H}_1\tilde{H}_1^* - I||_F$ | $||\tilde{H}_2\tilde{H}_2^* - I||_F$ | $||\tilde{H}_3\tilde{H}_3^* - I||_F$ |
|-----|-----------------------------------|-----------------------------------|-----------------------------------|
| 60  | $3.16 \times 10^{-14}$            | $3.09 \times 10^{-14}$           | $3.09 \times 10^{-14}$           |
| 80  | $5.70 \times 10^{-14}$            | $5.46 \times 10^{-14}$           | $5.46 \times 10^{-14}$           |
| 100 | $8.20 \times 10^{-14}$            | $7.88 \times 10^{-14}$           | $7.88 \times 10^{-14}$           |
| 120 | $1.10 \times 10^{-14}$            | $1.05 \times 10^{-14}$           | $1.05 \times 10^{-14}$           |
| 140 | $1.42 \times 10^{-14}$            | $1.36 \times 10^{-14}$           | $1.36 \times 10^{-14}$           |

Table 6.1: Symplecticity of Hermitian matrices

However, this does not solve the problem: after undergoing a similarity by $Q$ of type (6.13), (6.14), or (6.15), roundoff error may again cause the affected diagonal elements to become complex numbers with small imaginary parts. Thus the diagonal entries of $H$ have to be repeatedly adjusted. Our scheme for doing this is as follows:

- If $Q$ as in (6.13) is to be used, then we replace the $(p, p)$ and $(n + p, n + p)$ entries of $H$ by their real parts.
• If \( Q \) as in either (6.14) or (6.15) is to be used, then we replace the diagonal entries of \( H \) which will be effected by the active rotation by their real part. For example, if \( Q \) is of the form as in (6.15) and the active rotation is the outer rotation, then the \((p, p)\) and \((n + q, n + q)\) elements of \( H \) are replaced by their real parts.

Therefore, in both cases the \( 2 \times 2 \) target submatrix which is the form

\[
\begin{bmatrix}
a + i\delta_a & z \\
\bar{z} + \epsilon & b + i\delta_b
\end{bmatrix}, \quad a, b, \delta_a, \delta_b \in \mathbb{R}, \ z, \epsilon \in \mathbb{C}, |\epsilon| \ll |z|,
\]

is replaced by

\[
\begin{bmatrix}
a & z \\
\bar{z} + \epsilon & b
\end{bmatrix}.
\]

Further, in the computation of the \( 2 \times 2 \) rotation matrix \( R \) we only use the values \( a, b, \) and \( z, \) that is, we implicitly work on

\[
H_1 = \begin{bmatrix}
a & z \\
\bar{z} & b
\end{bmatrix}.
\]

Note that in practise we find that \(|\delta_a| \approx O(\epsilon)\) and \(|\delta_b| \approx O(\epsilon)|s|\).

From Proposition 6.1 we have that \( \tilde{H}_1 = R^*H_1R \) is diagonal if \( 2\alpha \equiv \phi \mod(\pi) \) and either \( \cos 2\theta = 0 \) whenever \( a = b, \) or \( \tan 2\theta = \frac{\delta}{b-a} \) whenever \( a \neq b. \)

In our work, we set \( \alpha = \frac{\pi}{2}. \) If \( a = b, \) then we set \((c, s) = \frac{1}{\sqrt{2}}(1, 1). \) On the other hand, if \( a \neq b, \) then \( \theta \) must satisfy the equation \( 2\theta = 2\tau/(b-a), \) that is,

\[
\tan \theta_{\text{small}} = \frac{\text{sign}(\tau)}{|\tau| + \sqrt{\tau^2 + 1}} \quad \text{where} \quad \tau = \frac{b-a}{2r}.
\]

Moreover, when \( a \neq b, \) choosing either \( \theta = \theta_{\text{small}} \) or \( \theta = \theta_{\text{large}} \) we obtain

\[
(c, s) = \left( \frac{1}{\sqrt{t^2 + 1}}, te \right) \quad \text{where} \quad t = \tan \theta.
\]

Note that as in the real case, using the small angle and the large angle makes a difference in the order of diagonal elements. That is, \( \theta_{\text{small}} \) preserves the order of \( a \) and \( b, \) whereas \( \theta_{\text{large}} \) has the effect of reversing their order.

The computation of \( Q \) as in (6.13)-(6.15) is given in the following.

**function**: \( Q = \text{Complex Rotation}(A, p, \text{sort}, t) \)

*Given a Hermitian matrix \( A \in \mathbb{C}^{2nx2n}, \) an integer \( p \) that satisfies \( 1 \leq p \leq n, \) a Boolean parameter "sort", and \( t > 0, \) this algorithm computes a complex symplectic plane rotation \( Q \in \mathbb{C}^{2nx2n} \) as in (6.13) such that if \( \tilde{A} = Q^*AQ, \) then \( \tilde{a}_{p,n+p} = \tilde{a}_{n+p,p} = 0. \) If \( \text{sort} = 0, \) then the order of the diagonal elements \((p, p)\)
and \((n+p,n+p)\) of \(A\) and \(\hat{A}\) is the same. If sort is nonzero, then the sorting angle is used for ordering the diagonal elements \((p,p)\) and \((n+p,n+p)\) of \(\hat{A}\) in decreasing order.

\[
\text{if } |a_{p,n+p}| > tol \text{ then} \quad \% \text{compute small angle}
\]

\[
\text{if } a_{pp} \neq a_{n+p,n+p} \text{ then}
\]

\[
\tau = \frac{a_{n+p,n+p} - a_{pp}}{2|a_{p,n+p}|}
\]

\[
t = \frac{\text{sign}(\tau)}{|\tau| + \sqrt{\tau^2 + 1}}
\]

\[
\text{if (sort \neq 0) and } (a_{pp} < a_{n+p,n+p}) \text{ then}
\]

\[
t = -\frac{1}{t} \quad \% \text{use large angle}
\]

\[
\text{endif}
\]

\[
c = \frac{1}{\sqrt{t^2 + 1}}; \quad s = tc
\]

\[
\text{else}
\]

\[
c = \frac{1}{\sqrt{2}}; \quad s = \frac{1}{\sqrt{2}}
\]

\[
\text{endif}
\]

\[
\alpha = \frac{\phi}{2} \quad \% \phi \text{ denotes the angle of } a_{p,n+p}
\]

\[
Q = I_{2n}; \quad Q(p,p) = e^{i\alpha}c; \quad Q(p,n+p) = e^{i\alpha}s
\]

\[
Q(n+p,p) = -e^{-i\alpha}s; \quad Q(n+p,n+p) = e^{-i\alpha}c
\]

\[
\text{else}
\]

\[
Q = I_{2n}
\]

\[
\text{endif}
\]

\textbf{function: } Q = \text{Complex Double}(A, p, q, sort, tol)

Given a Hermitian matrix \(A \in \mathbb{C}^{2n \times 2n}\), integers \(p\) and \(q\) that satisfy \(1 \leq p < q \leq n\), a Boolean parameter “sort”, and \(tol > 0\), this algorithm computes a complex symplectic double rotation \(Q \in \mathbb{C}^{2n \times 2n}\) as in (6.14) such that if \(\hat{A} = Q^*AQ\), then \(\hat{a}_{pq} = \hat{a}_{qp} = 0\). If sort = 0, then the order of the diagonal elements \((p,p)\) and \((q,q)\) of \(A\) and \(\hat{A}\) is the same. If sort is nonzero, then the sorting angle is used for ordering the diagonal elements \((p,p)\) and \((q,q)\) of \(\hat{A}\) in decreasing order.

\[
\text{if } |a_{pq}| > tol \text{ then} \quad \% \text{compute small angle}
\]

\[
\text{if } a_{pp} \neq a_{qq} \text{ then}
\]

\[
\tau = \frac{a_{qq} - a_{pp}}{2|a_{pq}|}
\]
\[
t = \frac{\text{sign } \tau}{|\tau| + \sqrt{\tau^2 + 1}}
\]

if \((\text{sort} \neq 0)\) and \((a_{pp} < a_{qq})\) then
\[
t = -\frac{1}{t}
\]
% use large angle
endif

\[
c = \frac{1}{\sqrt{t^2 + 1}}; \quad s = tc
\]
else
\[
c = \frac{1}{\sqrt{2}}; \quad s = \frac{1}{\sqrt{2}}
\]
endif

\[
\alpha = \frac{\phi}{2}
\]
% \(\phi\) denotes the angle of \(a_{pq}\)

\[
Q = I_{2n}; \quad Q(p, p) = e^{i\alpha_c}; \quad Q(p, q) = e^{i\alpha_s}
\]
\[
Q(q, p) = -e^{-i\alpha_s}; \quad Q(q, q) = e^{-i\alpha_c}
\]
\[
Q(n + p, n + p) = e^{i\alpha_c}; \quad Q(n + p, n + q) = e^{i\alpha_s}
\]
\[
Q(n + q, n + p) = -e^{-i\alpha_s}; \quad Q(n + q, n + q) = e^{-i\alpha_c}
\]
else
\[
Q = I_{2n}
\]
endif

Note that an algorithm for setting \(\hat{a}_{n+p,n+q} = \hat{a}_{n+q,n+p} = 0\), can be obtained by replacing \(p\) and \(q\) with \(n+p\) and \(n+q\), respectively in function Complex Double.

**function: Q = Complex Concentric**\((A, p, n + q, \text{sort}, \text{tol})\)

*Given a Hermitian matrix* \(A \in \mathbb{C}^{2n \times 2n}\), *integers* \(p\) and \(n + q\) that satisfy \(1 \leq p < q \leq n\), *a Boolean parameter* “sort”, and \(\text{tol} > 0\), *this algorithm computes a complex symplectic concentric rotation* \(Q \in \mathbb{C}^{2n \times 2n}\) *as in (6.15) such that if* \(\hat{A} = Q^*AQ\), *then* \(\hat{a}_{p,n+q} = \hat{a}_{n+q,p} = 0\). *If sort = 0, then the order of the diagonal elements* \((p, p)\) *and* \((n + q, n + q)\) *of* \(A\) *and* \(\hat{A}\) *is the same. If sort is nonzero, then the sorting angle is used for ordering the diagonal elements* \((p, p)\) *and* \((n + q, n + q)\) *of* \(\hat{A}\) *in decreasing order.*

if \(|a_{p,n+q}| > \text{tol}\) then
% compute small angle
if \(a_{pp} \neq a_{n+q,n+q}\) then
\[
\tau = \frac{a_{n+q,n+q} - a_{pp}}{2|a_{p,n+q}|}
\]
\[
t = \frac{\text{sign } \tau}{|\tau| + \sqrt{\tau^2 + 1}}
\]
if (sort \neq 0) \text{ and } (a_{pp} < a_{n+q,n+q}) \text{ then } \\
\quad t = \frac{1}{l} \quad \% \text{ use large angle} \\
\quad \text{endif} \\
\quad c = \frac{1}{\sqrt{t^2 + 1}}; \quad s = tc \\
\text{else} \\
\quad c = \frac{1}{\sqrt{2}}; \quad s = \frac{1}{\sqrt{2}} \\
\text{endif} \\
\quad \alpha = \frac{\phi}{2} \quad \% \phi \text{ denotes the angle of } a_{p,n+q} \\
\quad Q = I_{2n}; \quad Q(p,p) = e^{i\alpha}c; \quad Q(p,n+q) = e^{i\alpha}s \\
\quad Q(n+q,p) = -e^{-i\alpha}s; \quad Q(n+q,n+q) = e^{-i\alpha}c \\
\quad Q(q,q) = e^{i\alpha}c; \quad Q(q,n+p) = e^{i\alpha}s \\
\quad Q(n+p,q) = -e^{-i\alpha}s; \quad Q(n+p,n+p) = e^{-i\alpha}c \\
\text{else} \\
\quad Q = I_{2n} \\
\text{endif} \\

\begin{align*}
\end{align*}

Note that an algorithm for setting \( \hat{a}_{q,n+p} = \hat{a}_{n+p,q} = 0 \), can be obtained by replacing \( p \) and \( n+q \) with \( q \) and \( n+p \), respectively in function Complex Concentric.

We now present structure preserving algorithms for computing the structured spectral decomposition of a complex symplectic Hermitian matrix. As remarked earlier, we use the sweep designs \( S_4, S_5 \) and \( S_6 \). By replacing the functions Rotation, Double, and Concentric in the real Algorithms 7 and 9 with the functions Complex Rotation, Complex Double, and Complex Concentric, respectively, we have the desired algorithms for the complex case. For the convenience of the reader, these algorithms are given below. We will use \( CS_{4}, CS_{5} \) and \( CS_{6} \) to denote the complex algorithms using the small angle and strategies \( S_4, S_5 \) and \( S_6 \), respectively. \( CS_{4s}, CS_{5s} \) and \( CS_{6s} \) will denote the complex algorithms using strategies \( S_4, S_5 \) and \( S_6 \) with the sorting angle, respectively.

Pseudo-code for each algorithm is given below. A Boolean parameter "sort" is given to each algorithm. This parameter will determine the use of either the small angle or the sorting angle. If sort is zero, then the small angle is used by each function called by the algorithm. On the other hand, when the input parameter sort \neq 0, the sorting angle applies to an appropriate iteration as described in the real case. That is, for the algorithm \( CS_{4s} \), the sorting angle applies to every iteration, while it applies only when the pivots are in the strict upper triangular
part of $E$ and in upper off-diagonal block $F$ where $A = \begin{bmatrix} F & E \\ H & F \end{bmatrix}$ for the algorithms $CS_{5b}$ and $CS_{6b}$.

**Algorithm 13 ($CS_4, CS_{4b}$).** Given a complex symplectic Hermitian matrix $A \in \mathbb{C}^{2n \times 2n}$, a Boolean parameter "sort", $tol > 0$, and $reloff.tol > 0$, this algorithm overwrites $A$ with a diagonal matrix $\Sigma = R^*AR$ where $R$ is complex symplectic unitary and $reloff(A) \leq reloff.tol$. If "sort" is zero, then the small angle is used at every iteration. On the other hand, if "sort" is nonzero, then the sorting angle is used at every iteration and results in the first $n$ diagonal entries of $\Sigma$ appearing in decreasing order.

\[
R = I_{2n} \\
reloff(A) = \frac{\text{off}(A)}{\|A\|_F} \\
\text{while } reloff(A) > reloff.tol \\
\text{for } i = 1 : n \\
\quad A(i, i) = \text{Re}(A(i, i)); A(j, j) = \text{Re}(A(j, j)) \\
\quad Q = \text{Complex Rotation}(A, i, \text{sort}, tol) \\
\quad A = Q^*AQ; R = RQ \\
\text{endfor} \\
\text{for } i = 1 : n - 1 \\
\quad \text{for } j = i + 1 : n \\
\quad \quad A(i, i) = \text{Re}(A(i, i)); A(j, j) = \text{Re}(A(j, j)) \\
\quad \quad Q = \text{Complex Double}(A, i, j, \text{sort}, tol) \\
\quad \quad A = Q^*AQ; R = RQ \\
\quad \text{endfor} \\
\text{endfor} \\
\text{for } i = 1 : n - i \\
\quad \text{for } j = n + i + 1 : 2n \\
\quad \quad A(i, i) = \text{Re}(A(i, i)); A(j, j) = \text{Re}(A(j, j)) \\
\quad \quad Q = \text{Complex Concentric}(A, i, j, \text{sort}, tol) \\
\quad \quad A = Q^*AQ; R = RQ \\
\quad \text{endfor} \\
\text{endfor} \\
\text{reloff(A)} = \frac{\text{off}(A)}{\|A\|_F} \\
\text{endwhile}

**Algorithm 14 ($CS_5, CS_{5b}$).** Given a complex symplectic Hermitian matrix $A \in \mathbb{C}^{2n \times 2n}$, a Boolean parameter "sort", $tol > 0$, and $reloff.tol > 0$, this algorithm overwrites $A$ with $R^*AR$ where $R$ is complex symplectic unitary and $reloff(A) \leq reloff.tol$. If "sort" is zero, then the small angle is used at every iteration. On the other hand, if "sort" is nonzero, then the sorting angle is used when the pivot
element is in the first \( n \) rows, that is, row \( i = 1, 2, \ldots, n \), and results in the first \( n \) diagonal entries of \( \Sigma \) appearing in decreasing order.

\[
R = I_{2n}
\]

\[
\text{reloff}(A) = \frac{\text{off}(A)}{\|A\|_F}
\]

while \( \text{reloff}(A) > \text{reloff.tol} \)

for \( i = 1 : n \)

for \( j = i + 1 : 2n \)

if \( j \leq n \)

\[
A(i, i) = \text{Re}(A(i, i)); A(j, j) = \text{Re}(A(j, j))
\]

\[
Q = \text{Complex Double}(A, i, j, \text{sort}, \text{tol})
\]

\[
A = Q^*AQ; R = RQ
\]

elseif \( j = n + i \)

\[
A(i, i) = \text{Re}(A(i, i)); A(j, j) = \text{Re}(A(j, j))
\]

\[
Q = \text{Complex Rotation}(A, i, \text{sort}, \text{tol})
\]

\[
A = Q^*AQ; R = RQ
\]

else

\[
A(i, i) = \text{Re}(A(i, i)); A(j, j) = \text{Re}(A(j, j))
\]

\[
Q = \text{Complex Concentric}(A, i, j, \text{sort}, \text{tol})
\]

\[
A = Q^*AQ; R = RQ
\]

endif

endfor

endfor

for \( i = n + 1 : 2n - 1 \)

for \( j = i + 1 : 2n \)

\[
A(i, i) = \text{Re}(A(i, i)); A(j, j) = \text{Re}(A(j, j))
\]

% both strategies use small angle

\[
Q = \text{Complex Double}(A, i, j, 0, \text{tol})
\]

\[
A = Q^*AQ; R = RQ
\]

endfor

endfor

\[
\text{reloff}(A) = \frac{\text{off}(A)}{\|A\|_F}
\]

endwhile

**Algorithm 15** \((\text{CS}_6, \text{CS}_{6s})\). Given a complex symplectic Hermitian matrix \( A \in \mathbb{C}^{2n \times 2n} \), a Boolean parameter \( \text{"sort"}, \text{tol} > 0 \), and \( \text{reloff}.\text{tol} > 0 \), this algorithm overwrites \( A \) with \( R^*AR \) where \( R \) is complex symplectic unitary and \( \text{reloff}(A) \leq \text{reloff}.\text{tol} \). If \( \text{"sort"} \) is zero, then the small angle is used at every iteration. On the other hand, if \( \text{"sort"} \) is nonzero, then the sorting angle is used when the pivot element is in the first \( n \) rows, that is, row \( i = 1, 2, \ldots, n \), and results in the first \( n \) diagonal entries of \( \Sigma \) appearing in decreasing order.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
\[ R = I_{2n} \]
\[ \text{reloff}(A) = \frac{\text{off}(A)}{\|A\|_F} \]

while \( \text{reloff}(A) > \text{reloff.tol} \)

for \( i = 1 : n - 1 \)
  for \( j = i + 1 : n \)
    \[ A(i,i) = \text{Re}(A(i,i)); \quad A(j,j) = \text{Re}(A(j,j)) \]
    \[ Q = \text{Complex Double}(A, i, j, \text{sort}, \text{tol}) \]
    \[ A = Q^*AQ; \quad R = RQ \]
  endfor
endfor

for \( i = 1 : n \)
  \[ A(i,i) = \text{Re}(A(i,i)); \quad A(j,j) = \text{Re}(A(j,j)) \]
  \[ Q = \text{Complex Rotation}(A, i, \text{sort}, \text{tol}) \]
  \[ A = Q^*AQ; \quad R = RQ \]
endfor

for \( i = 1 : n - 1 \)
  for \( j = n + i + 1 : 2n \)
    \[ A(i,i) = \text{Re}(A(i,i)); \quad A(j,j) = \text{Re}(A(j,j)) \]
    \[ Q = \text{Complex Concentric}(A, i, j, \text{sort}, \text{tol}) \]
    \[ A = Q^*AQ; \quad R = RQ \]
  endfor
endfor

for \( j = n + 1 : 2n - 1 \)
  for \( i = j - n + 1 : n \)
    \[ A(i,i) = \text{Re}(A(i,i)); \quad A(j,j) = \text{Re}(A(j,j)) \]
    \[ Q = \text{Complex Concentric}(A, i, j, \text{sort}, \text{tol}) \]
    \[ A = Q^*AQ; \quad R = RQ \]
  endfor
endfor

for \( i = n + 1 : 2n - 1 \)
  for \( j = i + 1 : 2n \)
    \[ A(i,i) = \text{Re}(A(i,i)); \quad A(j,j) = \text{Re}(A(j,j)) \]
    \% both strategies use small angle
    \[ Q = \text{Complex Double}(A, i, j, 0, \text{tol}) \]
    \[ A = Q^*AQ; \quad R = RQ \]
  endfor
endfor

\[ \text{reloff}(A) = \frac{\text{off}(A)}{\|A\|_F} \]

endwhile
We now have the structure preserving algorithms for computing the structured spectral decomposition of $H$. The following section presents numerical experiments for computing a complex symplectic SVD of $A$.

### 6.1.2 Numerical Experiments

All experiments presented here comprise two steps: a step in which the complex symplectic unitary matrix $U$ is computed and a step in which the structured spectral decomposition of the complex symplectic Hermitian matrix $H$ is computed.

We point out that the algorithm in [21] generates complex symplectic matrices. However they are not random and unsuitable for numerical experiments. In our work, test matrices for all experiments were randomly generated complex symplectic matrices using MATLAB programs written by N. Mackey that encode a quaternion based algorithm given in [24].

The first set of experiments in this section investigates the number of iterations needed by Newton iteration as in (6.2) to make $X_{k+1}$ closer to $U$. The stopping criterion used for the Newton iteration was

$$
||X_{k+1} - X_k|| < 2 \times 10^{-14}.
$$

The Newton iteration was performed on 100 random complex symplectic matrices of dimension $2n \times 2n$ for $n = 30, 40, 50, 60, 70$ with condition number 10. Experimental results show that about 7 or 8 iterations at most were required for any test matrix.

The second set of experiments were performed to test all structured factors of the complex symplectic SVD of $A$ as in (6.1), and to see additional quantities of numerical interest. Fifty complex symplectic matrices of dimension $2n \times 2n$ for $n = 30, 40, 50, 60, 70$ with condition number 10 were generated. The Newton iteration was first performed on each of these matrices to compute the complex symplectic unitary $U$, and then the complex symplectic Hermitian factor $H$ was determined by $H = U^* A$. Each of these matrices $H$ was the input matrix to our algorithms described in Section 6.1.1. That is, all six algorithms, $CS_4$, $CS_5$, $CS_6$, $CS_8$, and $CS_{6s}$, were run on each of matrices $H$. All algorithms used $reloff.tol = 10^{-13} \|H\|_F$; the tolerance threshold for performing a Jacobi-like action was chosen as $tol = 10^{-14}$. The results we obtained are shown, on average, in Tables 6.2 - 6.7, and discussed below.

- **Sweeps** denotes the average number of sweeps needed by each algorithm to drop $reloff$ below $reloff.tol$. It can be seen from Tables 6.2 - 6.7 that the average number of sweeps increases with increasing matrix sizes for all algorithms. We remark that the standard deviation of the number of sweeps was consistently very low — between 0 and 0.5. It is obvious that the average number of sweeps for algorithms $CS_4$ and $CS_{4s}$ is higher than that of the
Table 6.2: Statistical data for the complex symplectic SVD: CS₄

<table>
<thead>
<tr>
<th>2n</th>
<th>60</th>
<th>80</th>
<th>100</th>
<th>120</th>
<th>140</th>
</tr>
</thead>
<tbody>
<tr>
<td>sweeps</td>
<td>8.70</td>
<td>9.05</td>
<td>9.14</td>
<td>9.75</td>
<td>9.88</td>
</tr>
<tr>
<td>reloff</td>
<td>1.80 \times 10^{-13}</td>
<td>3.72 \times 10^{-14}</td>
<td>7.65 \times 10^{-14}</td>
<td>1.32 \times 10^{-13}</td>
<td>1.52 \times 10^{-13}</td>
</tr>
<tr>
<td>|AA^*-I|_F</td>
<td>1.88 \times 10^{-14}</td>
<td>2.54 \times 10^{-14}</td>
<td>3.05 \times 10^{-14}</td>
<td>3.61 \times 10^{-14}</td>
<td>4.24 \times 10^{-14}</td>
</tr>
<tr>
<td>|UU^*-I|_F</td>
<td>2.45 \times 10^{-14}</td>
<td>2.72 \times 10^{-14}</td>
<td>3.99 \times 10^{-14}</td>
<td>3.03 \times 10^{-14}</td>
<td>3.31 \times 10^{-14}</td>
</tr>
<tr>
<td>|UU^*-I|_F</td>
<td>1.18 \times 10^{-14}</td>
<td>1.24 \times 10^{-14}</td>
<td>1.85 \times 10^{-14}</td>
<td>1.25 \times 10^{-14}</td>
<td>1.29 \times 10^{-14}</td>
</tr>
<tr>
<td>|HH^*-I|_F</td>
<td>3.08 \times 10^{-14}</td>
<td>4.08 \times 10^{-14}</td>
<td>5.56 \times 10^{-14}</td>
<td>5.39 \times 10^{-14}</td>
<td>6.21 \times 10^{-14}</td>
</tr>
<tr>
<td>|H - H^*|_F</td>
<td>2.42 \times 10^{-15}</td>
<td>2.20 \times 10^{-15}</td>
<td>2.61 \times 10^{-15}</td>
<td>1.69 \times 10^{-15}</td>
<td>1.51 \times 10^{-15}</td>
</tr>
<tr>
<td>|WW^*-I|_F</td>
<td>7.92 \times 10^{-14}</td>
<td>1.23 \times 10^{-13}</td>
<td>1.79 \times 10^{-13}</td>
<td>2.30 \times 10^{-13}</td>
<td>2.94 \times 10^{-13}</td>
</tr>
<tr>
<td>|WW^*-I|_F</td>
<td>7.93 \times 10^{-14}</td>
<td>1.19 \times 10^{-13}</td>
<td>1.73 \times 10^{-13}</td>
<td>2.26 \times 10^{-13}</td>
<td>2.90 \times 10^{-13}</td>
</tr>
<tr>
<td>|QQ^*-I|_F</td>
<td>7.53 \times 10^{-14}</td>
<td>1.17 \times 10^{-13}</td>
<td>1.71 \times 10^{-13}</td>
<td>2.25 \times 10^{-13}</td>
<td>2.88 \times 10^{-13}</td>
</tr>
<tr>
<td>|QQ^*-I|_F</td>
<td>7.53 \times 10^{-14}</td>
<td>1.18 \times 10^{-13}</td>
<td>1.71 \times 10^{-13}</td>
<td>2.25 \times 10^{-13}</td>
<td>2.88 \times 10^{-13}</td>
</tr>
<tr>
<td>diag.alg</td>
<td>2.98 \times 10^{-14}</td>
<td>3.80 \times 10^{-14}</td>
<td>4.82 \times 10^{-14}</td>
<td>5.85 \times 10^{-14}</td>
<td>6.91 \times 10^{-14}</td>
</tr>
<tr>
<td>diag,Matlab</td>
<td>3.48 \times 10^{-15}</td>
<td>4.07 \times 10^{-15}</td>
<td>4.28 \times 10^{-15}</td>
<td>4.89 \times 10^{-15}</td>
<td>4.81 \times 10^{-15}</td>
</tr>
<tr>
<td>rel.eig</td>
<td>1.67 \times 10^{-14}</td>
<td>2.11 \times 10^{-14}</td>
<td>2.67 \times 10^{-14}</td>
<td>3.17 \times 10^{-14}</td>
<td>3.72 \times 10^{-14}</td>
</tr>
<tr>
<td>rel.singular</td>
<td>1.60 \times 10^{-14}</td>
<td>2.08 \times 10^{-14}</td>
<td>2.61 \times 10^{-14}</td>
<td>3.12 \times 10^{-14}</td>
<td>3.67 \times 10^{-14}</td>
</tr>
</tbody>
</table>

Table 6.3: Statistical data for the complex symplectic SVD: CS₄₈

<table>
<thead>
<tr>
<th>2n</th>
<th>60</th>
<th>80</th>
<th>100</th>
<th>120</th>
<th>140</th>
</tr>
</thead>
<tbody>
<tr>
<td>sweeps</td>
<td>7.40</td>
<td>7.95</td>
<td>8.06</td>
<td>8.20</td>
<td>8.34</td>
</tr>
<tr>
<td>reloff</td>
<td>1.01 \times 10^{-13}</td>
<td>1.13 \times 10^{-13}</td>
<td>1.77 \times 10^{-13}</td>
<td>1.82 \times 10^{-14}</td>
<td>1.12 \times 10^{-15}</td>
</tr>
<tr>
<td>|AA^*-I|_F</td>
<td>1.88 \times 10^{-14}</td>
<td>2.54 \times 10^{-14}</td>
<td>3.05 \times 10^{-14}</td>
<td>3.61 \times 10^{-14}</td>
<td>4.24 \times 10^{-14}</td>
</tr>
<tr>
<td>|UU^*-I|_F</td>
<td>2.45 \times 10^{-14}</td>
<td>2.72 \times 10^{-14}</td>
<td>3.99 \times 10^{-14}</td>
<td>3.03 \times 10^{-14}</td>
<td>3.31 \times 10^{-14}</td>
</tr>
<tr>
<td>|UU^*-I|_F</td>
<td>1.18 \times 10^{-14}</td>
<td>1.24 \times 10^{-14}</td>
<td>1.85 \times 10^{-14}</td>
<td>1.25 \times 10^{-14}</td>
<td>1.29 \times 10^{-14}</td>
</tr>
<tr>
<td>|HH^*-I|_F</td>
<td>3.08 \times 10^{-14}</td>
<td>4.08 \times 10^{-14}</td>
<td>5.56 \times 10^{-14}</td>
<td>5.39 \times 10^{-14}</td>
<td>6.21 \times 10^{-14}</td>
</tr>
<tr>
<td>|H - H^*|_F</td>
<td>2.42 \times 10^{-15}</td>
<td>2.20 \times 10^{-15}</td>
<td>2.61 \times 10^{-15}</td>
<td>1.69 \times 10^{-15}</td>
<td>1.51 \times 10^{-15}</td>
</tr>
<tr>
<td>|WW^*-I|_F</td>
<td>7.77 \times 10^{-14}</td>
<td>1.23 \times 10^{-13}</td>
<td>1.70 \times 10^{-13}</td>
<td>2.27 \times 10^{-13}</td>
<td>2.90 \times 10^{-13}</td>
</tr>
<tr>
<td>|WW^*-I|_F</td>
<td>7.67 \times 10^{-14}</td>
<td>1.21 \times 10^{-13}</td>
<td>1.68 \times 10^{-13}</td>
<td>2.25 \times 10^{-13}</td>
<td>2.88 \times 10^{-13}</td>
</tr>
<tr>
<td>|QQ^*-I|_F</td>
<td>7.68 \times 10^{-14}</td>
<td>1.21 \times 10^{-13}</td>
<td>1.68 \times 10^{-13}</td>
<td>2.25 \times 10^{-13}</td>
<td>2.88 \times 10^{-13}</td>
</tr>
<tr>
<td>diag.alg</td>
<td>2.85 \times 10^{-14}</td>
<td>3.86 \times 10^{-14}</td>
<td>4.67 \times 10^{-14}</td>
<td>5.61 \times 10^{-14}</td>
<td>6.35 \times 10^{-14}</td>
</tr>
<tr>
<td>diag,Matlab</td>
<td>3.48 \times 10^{-15}</td>
<td>4.07 \times 10^{-15}</td>
<td>4.28 \times 10^{-15}</td>
<td>4.89 \times 10^{-15}</td>
<td>4.81 \times 10^{-15}</td>
</tr>
<tr>
<td>rel.eig</td>
<td>1.69 \times 10^{-14}</td>
<td>2.16 \times 10^{-14}</td>
<td>2.62 \times 10^{-14}</td>
<td>3.04 \times 10^{-14}</td>
<td>3.59 \times 10^{-14}</td>
</tr>
<tr>
<td>rel.singular</td>
<td>1.57 \times 10^{-14}</td>
<td>2.10 \times 10^{-14}</td>
<td>2.55 \times 10^{-14}</td>
<td>3.03 \times 10^{-14}</td>
<td>3.43 \times 10^{-14}</td>
</tr>
<tr>
<td>2n</td>
<td>sweeps</td>
<td>60</td>
<td>80</td>
<td>100</td>
<td>120</td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td></td>
<td>reloff</td>
<td>6.17 × 10⁻¹⁴</td>
<td>1.13 × 10⁻¹³</td>
<td>8.12 × 10⁻¹⁴</td>
<td>1.16 × 10⁻¹³</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>AA* − I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UU* − I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UU* − I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>HH* − I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>H − H*</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>WW* − I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>WW* − I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>QQ* − I</td>
<td></td>
<td>_F</td>
</tr>
<tr>
<td></td>
<td>diag.alg</td>
<td>3.91 × 10⁻¹⁴</td>
<td>5.24 × 10⁻¹⁴</td>
<td>6.73 × 10⁻¹⁴</td>
<td>8.26 × 10⁻¹⁴</td>
</tr>
<tr>
<td></td>
<td>diag.Matlab</td>
<td>3.48 × 10⁻¹⁵</td>
<td>4.07 × 10⁻¹⁵</td>
<td>4.28 × 10⁻¹⁵</td>
<td>4.89 × 10⁻¹⁵</td>
</tr>
<tr>
<td></td>
<td>releig</td>
<td>2.18 × 10⁻¹⁴</td>
<td>2.88 × 10⁻¹⁴</td>
<td>3.66 × 10⁻¹⁴</td>
<td>4.44 × 10⁻¹⁴</td>
</tr>
<tr>
<td></td>
<td>rel.singular</td>
<td>2.14 × 10⁻¹⁴</td>
<td>2.82 × 10⁻¹⁴</td>
<td>3.66 × 10⁻¹⁴</td>
<td>4.35 × 10⁻¹⁴</td>
</tr>
</tbody>
</table>

Table 6.4: Statistical data for the complex symplectic SVD: CS₅₆

<table>
<thead>
<tr>
<th>2n</th>
<th>sweeps</th>
<th>60</th>
<th>80</th>
<th>100</th>
<th>120</th>
<th>140</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>reloff</td>
<td>6.71 × 10⁻¹⁴</td>
<td>7.61 × 10⁻¹⁴</td>
<td>7.87 × 10⁻¹⁴</td>
<td>1.31 × 10⁻¹³</td>
<td>3.87 × 10⁻¹⁴</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>AA* − I</td>
<td></td>
<td>_F</td>
<td>1.88 × 10⁻¹⁴</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UU* − I</td>
<td></td>
<td>_F</td>
<td>2.45 × 10⁻¹⁴</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UU* − I</td>
<td></td>
<td>_F</td>
<td>1.23 × 10⁻¹³</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>HH* − I</td>
<td></td>
<td>_F</td>
<td>3.08 × 10⁻¹⁴</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>H − H*</td>
<td></td>
<td>_F</td>
<td>2.42 × 10⁻¹⁵</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>WW* − I</td>
<td></td>
<td>_F</td>
<td>8.83 × 10⁻¹⁴</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>WW* − I</td>
<td></td>
<td>_F</td>
<td>8.58 × 10⁻¹⁴</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>QQ* − I</td>
<td></td>
<td>_F</td>
<td>8.49 × 10⁻¹⁴</td>
</tr>
<tr>
<td></td>
<td>diag.alg</td>
<td>3.07 × 10⁻¹⁴</td>
<td>4.20 × 10⁻¹⁴</td>
<td>5.30 × 10⁻¹⁴</td>
<td>6.33 × 10⁻¹⁴</td>
<td>7.67 × 10⁻¹⁴</td>
</tr>
<tr>
<td></td>
<td>diag.Matlab</td>
<td>3.48 × 10⁻¹⁵</td>
<td>4.07 × 10⁻¹⁵</td>
<td>4.28 × 10⁻¹⁵</td>
<td>4.89 × 10⁻¹⁵</td>
<td>4.81 × 10⁻¹⁵</td>
</tr>
<tr>
<td></td>
<td>releig</td>
<td>1.79 × 10⁻¹⁴</td>
<td>2.32 × 10⁻¹⁴</td>
<td>2.96 × 10⁻¹⁴</td>
<td>3.40 × 10⁻¹⁴</td>
<td>4.05 × 10⁻¹⁴</td>
</tr>
<tr>
<td></td>
<td>rel.singular</td>
<td>1.69 × 10⁻¹⁴</td>
<td>2.29 × 10⁻¹⁴</td>
<td>2.91 × 10⁻¹⁴</td>
<td>3.38 × 10⁻¹⁴</td>
<td>4.04 × 10⁻¹⁴</td>
</tr>
</tbody>
</table>

Table 6.5: Statistical data for the complex symplectic SVD: CS₅₈
<table>
<thead>
<tr>
<th>2n sweeps</th>
<th>60</th>
<th>80</th>
<th>100</th>
<th>120</th>
<th>140</th>
</tr>
</thead>
<tbody>
<tr>
<td>reloff</td>
<td>$4.82 \times 10^{-14}$</td>
<td>$2.93 \times 10^{-14}$</td>
<td>$2.58 \times 10^{-14}$</td>
<td>$3.17 \times 10^{-14}$</td>
<td>$3.97 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|AA^* - I|_F$</td>
<td>$1.88 \times 10^{-14}$</td>
<td>$2.54 \times 10^{-14}$</td>
<td>$3.05 \times 10^{-14}$</td>
<td>$3.61 \times 10^{-14}$</td>
<td>$4.24 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|UU^* - I|_F$</td>
<td>$2.45 \times 10^{-14}$</td>
<td>$2.72 \times 10^{-14}$</td>
<td>$3.99 \times 10^{-14}$</td>
<td>$3.03 \times 10^{-14}$</td>
<td>$3.31 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|UU^* - I|_F$</td>
<td>$1.18 \times 10^{-14}$</td>
<td>$1.24 \times 10^{-14}$</td>
<td>$1.85 \times 10^{-14}$</td>
<td>$1.25 \times 10^{-14}$</td>
<td>$1.29 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|HH^* - I|_F$</td>
<td>$3.08 \times 10^{-14}$</td>
<td>$4.08 \times 10^{-14}$</td>
<td>$5.56 \times 10^{-14}$</td>
<td>$5.39 \times 10^{-14}$</td>
<td>$6.21 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|HH^* - I|_F$</td>
<td>$2.42 \times 10^{-15}$</td>
<td>$2.20 \times 10^{-15}$</td>
<td>$2.61 \times 10^{-15}$</td>
<td>$1.69 \times 10^{-15}$</td>
<td>$1.51 \times 10^{-15}$</td>
</tr>
<tr>
<td>$|WW^* - I|_F$</td>
<td>$9.20 \times 10^{-14}$</td>
<td>$1.55 \times 10^{-13}$</td>
<td>$3.30 \times 10^{-13}$</td>
<td>$4.52 \times 10^{-13}$</td>
<td>$5.98 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|WW^* - I|_F$</td>
<td>$8.96 \times 10^{-14}$</td>
<td>$1.49 \times 10^{-13}$</td>
<td>$2.15 \times 10^{-13}$</td>
<td>$2.92 \times 10^{-13}$</td>
<td>$3.65 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|QQ^* - I|_F$</td>
<td>$8.87 \times 10^{-14}$</td>
<td>$1.46 \times 10^{-13}$</td>
<td>$2.12 \times 10^{-13}$</td>
<td>$2.88 \times 10^{-13}$</td>
<td>$3.59 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|QQ^* - I|_F$</td>
<td>$8.88 \times 10^{-14}$</td>
<td>$1.47 \times 10^{-13}$</td>
<td>$2.12 \times 10^{-13}$</td>
<td>$2.88 \times 10^{-13}$</td>
<td>$3.59 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$3.22 \times 10^{-14}$</td>
<td>$4.37 \times 10^{-14}$</td>
<td>$5.66 \times 10^{-14}$</td>
<td>$6.92 \times 10^{-14}$</td>
<td>$7.82 \times 10^{-14}$</td>
</tr>
<tr>
<td>diag,Matlab</td>
<td>$3.48 \times 10^{-15}$</td>
<td>$4.07 \times 10^{-15}$</td>
<td>$4.28 \times 10^{-15}$</td>
<td>$4.89 \times 10^{-15}$</td>
<td>$4.81 \times 10^{-15}$</td>
</tr>
<tr>
<td>releig</td>
<td>$1.84 \times 10^{-14}$</td>
<td>$2.41 \times 10^{-14}$</td>
<td>$3.10 \times 10^{-14}$</td>
<td>$3.74 \times 10^{-14}$</td>
<td>$4.12 \times 10^{-14}$</td>
</tr>
<tr>
<td>rel.singular</td>
<td>$1.78 \times 10^{-14}$</td>
<td>$2.38 \times 10^{-14}$</td>
<td>$3.05 \times 10^{-14}$</td>
<td>$3.70 \times 10^{-14}$</td>
<td>$4.16 \times 10^{-14}$</td>
</tr>
</tbody>
</table>

Table 6.6: Statistical data for the complex symplectic SVD: $CS_6$
other four algorithms for all matrix sizes under consideration. However, we remind the reader that the total cost per full sweep of CS4 and CS4s is approximately half of that of CS5, CS5s, CS6, and CS6s as discussed in the real case. Moreover, Tables 6.2 - 6.7 give additional information. That is, the average number of sweeps needed by an algorithm using the sorting angle is lower than that by a corresponding algorithm using the small angle for all dimensions under consideration. In fact, CS4, CS5, and CS6 required more sweeps than CS4s, CS5s, and CS6s, respectively.

- For all algorithms, reloff(H) in practice converges quadratically. The typical behavior of reloff(H) for a 140 × 140 complex symplectic Hermitian matrix is shown in Figure 6.1.

- The symplecticity of the test matrix A was tested via \( \| AA^* - I \|_F \). It can be seen from Tables 6.2 - 6.7 that this quantity was, on average, of the order \( 10^{-14} \), so our algorithms were tested on well structured matrices.

- \( \| UU^* - I \|_F \) and \( \| UU^* - I \|_F \) measure the deviation of the matrix U computed by Newton iteration (6.2) from symplecticity and unitary, respectively. Tables 6.2 - 6.7 show the average of these values taken over all 50 test matrices. As we see, the computed matrices U were both complex symplectic and unitary to within \( 3.31 \times 10^{-14} \). The departure from being complex symplectic and Hermitian of the factor \( H = U^* A \) is measured via \( \| HH^* - I \|_F \) and \( \| H - H^* \|_F \), respectively. We see that the computed matrices H were both complex symplectic as well as Hermitian to within order \( 10^{-14} \).

- Each algorithm computes W and Q which are approximations to the complex symplectic unitary matrices encoding the left and right singular vectors of the test matrix, respectively. In order to see how well the computed W and Q have both structures, \( \| WW^* - I \|_F \), \( \| WW^* - I \|_F \) and \( \| QQ^* - I \|_F \), \( \| QQ^* - I \|_F \) were calculated. As can be seen from Tables 6.2 - 6.7, for the matrices W and Q computed by all six algorithms, these quantities were, on average, of the order \( 10^{-13} \).

- The row labeled diag.alg reports the deviation of the computed matrix \( \Sigma \) from being real symplectic diagonal, that is, the deviation from pairing structure \((\sigma, 1/\sigma)\) between the singular values computed by our algorithms. Here diag.alg = \( \max \left| \text{diag}(S_1) \cdot \text{diag}(S_2) - 1 \right| \) where \( S_1 = \Sigma(1 : n, 1 : n) \), \( S_2 = \Sigma(n + 1 : 2n, n + 1 : 2n) \), and \( \cdot \) denotes array multiplication (as in MATLAB). Similarly, the row labeled diag.Matlab reports the deviation from pairing structure of the singular values computed by MATLAB’s svd function. If \( v \) denotes the vector of sorted singular values computed by svd(A), and \( x := v(1 : n), y := v(2n : -1 : n + 1) \), then diag.Matlab = \( \max |x \cdot y - 1| \). As
Figure 6.1: Typical convergence behavior of a $140 \times 140$ complex symplectic Hermitian matrix. Note that a full sweep of $CS_4$ and $CS_{6s}$ has roughly half as many iterations as a full sweep of other algorithms.
we see from Tables 6.2 - 6.7, the loss of the symplectic structure of $\Sigma$ was, on average, of the order $10^{-14}$ while the loss of pairing structure of singular values computed by MATLAB was of the order $10^{-15}$ for all algorithms and all dimensions under consideration.

- The last two rows of Tables 6.2 - 6.7 report the maximum relative error in the computed eigenvalues of $H$ and the maximum relative error in the computed singular values of $A$, respectively. Theoretically, the eigenvalues of $H$ and the singular values of $A$ are the same. To measure these maximum relative errors, we sorted the computed eigenvalues of $H$ and the computed singular values of $A$ and compared them with the sorted eigenvalues obtained by MATLAB’s $eig$ function and the sorted singular values obtained by MATLAB’s $svd$ function. The row $releig$ shows the maximum relative error of the computed eigenvalues and the row $rel.singular$ reports the maximum relative error in the computed singular values. Recall that

$$releig = \max_j \frac{|\lambda_j^{\text{eig}} - \lambda_j^{\text{alg}}|}{|\lambda_j^{\text{eig}}|},$$

where $\{\lambda_j^{\text{eig}}\}$ and $\{\lambda_j^{\text{alg}}\}$ denote the eigenvalues computed by MATLAB’s $eig$ function and by our algorithms, respectively, and

$$rel.singular = \max_j \frac{|\sigma_j^{\text{mat}} - \sigma_j^{\text{alg}}|}{|\sigma_j^{\text{mat}}|},$$

where $\{\sigma_j^{\text{mat}}\}$ and $\{\sigma_j^{\text{alg}}\}$ denote the singular values computed by MATLAB’s $svd$ function and by our algorithms, respectively. We see that the eigenvalues of $H$ and the singular values of $A$ computed by our strategies were comparable to MATLAB up to order $10^{-14}$.

A third set of experiments was designed to see how accurately our algorithms computed the singular values of a complex symplectic test matrix with pre-chosen singular values. Then we compared the chosen singular values to singular values computed by our strategies and to those computed by MATLAB’s $svd$ function. Some of the experiments are shown below.

**Example 14.** A complex symplectic matrix $A$ was generated with singular values $5, 5, 3, \frac{1}{5}, \frac{1}{5}, \frac{1}{3}$. In this case $A$ is well-conditioned with condition number 25, and has two set of repeated singular values of multiplicity 2. Table 6.8 shows the exact singular values and the relative errors in the singular value approximations computed by our algorithms and by MATLAB’s $svd$ function.
Table 6.8: Accuracy of the singular values 5, 5, 3, $\frac{1}{5}$, $\frac{1}{5}$, $\frac{1}{3}$

<table>
<thead>
<tr>
<th>$\sigma_i$</th>
<th>$CS_4$</th>
<th>$CS_{4a}$</th>
<th>$CS_5$</th>
<th>$CS_{5a}$</th>
<th>$CS_6$</th>
<th>$CS_{6a}$</th>
<th>Matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/5</td>
<td>8.7e-16</td>
<td>3.1e-16</td>
<td>4.4e-15</td>
<td>1.3e-15</td>
<td>3.1e-16</td>
<td>4.4e-16</td>
<td>2.0e-15</td>
</tr>
<tr>
<td>1/5</td>
<td>1.6e-15</td>
<td>2.4e-15</td>
<td>2.8e-15</td>
<td>1.8e-15</td>
<td>2.6e-15</td>
<td>2.1e-15</td>
<td>2.5e-15</td>
</tr>
<tr>
<td>1/3</td>
<td>7.6e-16</td>
<td>1.1e-15</td>
<td>2.9e-16</td>
<td>1.6e-15</td>
<td>3.3e-16</td>
<td>1.1e-15</td>
<td>6.6e-16</td>
</tr>
<tr>
<td>3</td>
<td>4.4e-16</td>
<td>7.4e-16</td>
<td>7.4e-16</td>
<td>4.4e-16</td>
<td>2.9e-16</td>
<td>3.3e-16</td>
<td>0.0</td>
</tr>
<tr>
<td>5</td>
<td>8.9e-16</td>
<td>2.2e-16</td>
<td>2.6e-15</td>
<td>3.9e-16</td>
<td>3.5e-16</td>
<td>3.9e-16</td>
<td>1.0e-15</td>
</tr>
<tr>
<td>5</td>
<td>5.8e-16</td>
<td>7.3e-16</td>
<td>5.3e-16</td>
<td>8.9e-16</td>
<td>4.4e-16</td>
<td>7.8e-16</td>
<td>7.1e-16</td>
</tr>
</tbody>
</table>

Table 6.9: Accuracy of the singular values near 3 and $\frac{1}{3}$

Example 15. A complex symplectic matrix $A$ was generated with singular values $a, b, c, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}$ where $[a, b, c] = [3, 3, 3] + [0.4, 2.7, 9.3] \times 10^{-15}$. Here $A$ is well-conditioned and has two clusters of singular values of “multiplicity” 3 — singular values near 3 and near $\frac{1}{3}$. The exact singular values and the relative errors are given in Table 6.9.

Example 16. A 10×10 complex symplectic matrix $A$ was generated with singular values $5, 5, \frac{1}{5}, \frac{1}{5}, a, b, c, \frac{1}{a}, \frac{1}{b}, \frac{1}{c}$ where $[a, b, c] = [3, 3, 3] + [1.3, 4.4, 6.7] \times 10^{-15}$. In this case, $A$ is a well-conditioned matrix with condition number 25, and has two set of repeated singular values of multiplicity 2 and two clusters of singular values of “multiplicity” 3. The exact singular values and the relative errors are reported in Table 6.10.

Example 17. A 20×20 complex symplectic matrix $A$ was generated with singular values $2, \frac{1}{2}, 4, \frac{1}{4}, 6, \frac{1}{6}, 7, \frac{1}{7}, a, b, c, \frac{1}{a}, \frac{1}{b}, \frac{1}{c}, x, y, z, \frac{1}{x}, \frac{1}{y}, \frac{1}{z}$ where $[a, b, c] = [3, 3, 3] + [1.3, 3.6, 6.2] \times 10^{-15}$ and $[x, y, z] = [5, 5, 5] + [3.6, 7.1, 8.0] \times 10^{-15}$. In this case, $A$ is well-conditioned and has 4 clusters of singular values of “multiplicity” 3 — singular values near 3, $\frac{1}{3}$, 5, and $\frac{1}{5}$. Table 6.11 reports the selected singular values and the relative errors in the singular value computed by our algorithms and by MATLAB’s svd function in each set of clusters.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
The results from Examples 14 - 17 as shown in Tables 6.8 - 6.11 show that our algorithms compute singular values with the relative error comparable to MATLAB's `svd` function.

### 6.2 One-sided Jacobi Method

Let $A$ be a $2n \times 2n$ complex symplectic matrix. By Theorem 1.5, a complex symplectic SVD exists. That is,

$$A = U\Sigma V^*, \quad (6.18)$$
where $U$ and $V$ are complex symplectic unitary and $\Sigma$ is real symplectic diagonal of the form

$$
\Sigma = \begin{bmatrix}
\Sigma_1 & 0 \\
0 & \Sigma_1^{-1}
\end{bmatrix}, \text{ where } \Sigma_1 = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n).
$$

In order to preserve the complex symplectic structure and to achieve a complex symplectic SVD of $A$ by using a one-sided Jacobi method, our tools are limited to matrices from $SpU(2n, \mathbb{C})$ as given in (1.12).

Once again, since a one-sided Jacobi method for the SVD of $A$ implicitly acts on $A^*A$ by a two-sided action, the tools in Section 6.1.1 — $2n \times 2n$ complex symplectic plane rotations as in (6.13), $2n \times 2n$ complex symplectic double rotations as in (6.14), and $2n \times 2n$ complex symplectic concentric rotations as in (6.15) — can be used. For our convenience, we will refer to these tools as one-sided complex rotations, one-sided complex double rotations, and one-sided complex concentric rotations, respectively. Algorithms for computing these tools are given at the end of this section. We will use sweep designs $S_4$, $S_5$ and $S_6$, for our algorithm.

As in the real case, the criterion for setting the off-diagonal elements $(p, q)$ and $(q, p)$ of $X = [x_{ij}] = A^*A$ to zero is defined as in (5.4), and, for convenience of the reader, reproduced here:

$$
\frac{|x_{pq}|}{\sqrt{x_{pp}x_{qq}}} > \text{threshold}, \quad (6.19)
$$

where threshold is defined by user. Here we set threshold $= 2n \times \text{eps}$. That is, the orthogonalization of the column pair $(p, q)$ is skipped if the reverse inequality in (6.19) holds.

We now present one-sided Jacobi-like algorithms for computing a complex symplectic SVD of a complex symplectic matrix $A$. We will use $OcS_4$, $OcS_5$ and $OcS_6$ to denote the one-sided Jacobi algorithms using the small angle and strategies $S_4$, $S_5$ and $S_6$, respectively. $OcS_4s$, $OcS_5s$ and $OcS_6s$ will denote the one-sided Jacobi algorithms using strategies $S_4$, $S_5$ and $S_6$ with the sorting angle, respectively.

Pseudo-code for each algorithm is given below. Again, a Boolean parameter “sort” is given to each algorithm to determine the use of either the small angle or the sorting angle. That is, if sort is zero, then we use the small angle inside each function called by the algorithm. On the other hand, when the input parameter sort $\neq 0$, the sorting angle applies to an appropriate iteration as described in the real case. That is, for the algorithm $OcS_4s$, the sorting angle applies to every iteration, while it applies only when rotation index pairs $(p, q)$ are in the strict upper triangular part of $E$ and in upper off-diagonal block $F$ where $A = [F_{ij}]$ for the algorithms $OcS_5s$ and $OcS_6s$. The parameter “count” keeps track of the number of iterations which are skipped. Pseudo-code for the function One-sided Complex Rotation, One-sided Complex Double, and One-sided Complex

---

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Concentric called by Algorithms 16 and 18 are given later at the end of this section.

The use of the sorting angle results in the first \( n \) diagonal entries of \( \Sigma \) appearing, at convergence, in decreasing order on the diagonal.

**Algorithm 16** (\( \text{OC}_4 \), \( \text{CS}_4 \)). Given a complex symplectic matrix \( A \in \mathbb{C}^{2n \times 2n} \), a Boolean parameter “sort”, and \( \text{tol} > 0 \), this algorithm computes the singular values \( \sigma_i \), and complex symplectic unitary matrices \( U \) and \( V \) so that \( A = U \Sigma V^* \) where \( \Sigma = \text{diag}(\sigma_i) \) is real symplectic diagonal. If “sort” is zero, then the first \( n \) diagonal entries of \( \Sigma \) do not appear in any particular order on the diagonal. On the other hand, if “sort” is nonzero, then the first \( n \) diagonal entries of \( \Sigma \) appearing in decreasing order.

\[
V = I_{2n}
\]

while count < \( n^2 \)

\[
\begin{align*}
& \text{for } i = 1 : n \\
& \quad Q = \text{One-sided Complex Rotation}(A, i, \text{sort}, \text{tol}) \\
& \quad A = AQ; V = VQ \\
& \text{endfor} \\
& \text{for } i = 1 : n - 1 \\
& \quad \text{for } j = i + 1 : n \\
& \quad \quad Q = \text{One-side Complex Double}(A, i, j, \text{sort}, \text{tol}) \\
& \quad \quad A = AQ; V = VQ \\
& \quad \text{endfor} \\
& \text{endfor} \\
& \text{for } i = 1 : n - i \\
& \quad \text{for } j = n + i + 1 : 2n \\
& \quad \quad Q = \text{One-sided Complex Concentric}(A, i, j, \text{sort}, \text{tol}) \\
& \quad \quad A = AQ; V = VQ \\
& \quad \text{endfor} \\
& \text{endfor} \\
\end{align*}
\]

endwhile

\[
\begin{align*}
& \text{for } i = 1 : 2n \\
& \quad \sigma_i = \|A(:,i)\|; U(:,i) = \frac{A(:,i)}{\sigma_i} \\
& \text{endfor}
\end{align*}
\]

**Algorithm 17** (\( \text{OC}_5 \), \( \text{CS}_8 \)). Given a complex symplectic matrix \( A \in \mathbb{C}^{2n \times 2n} \), a Boolean parameter “sort”, and \( \text{tol} > 0 \), this algorithm computes the singular values \( \sigma_i \), and complex symplectic unitary matrices \( U \) and \( V \) so that \( A = U \Sigma V^* \) where \( \Sigma = \text{diag}(\sigma_i) \) is real symplectic diagonal. If “sort” is zero, then the first \( n \) diagonal entries of \( \Sigma \) do not appear in any particular order on the diagonal.
On the other hand, if "sort" is nonzero, then the first $n$ diagonal entries of $\Sigma$ appearing in decreasing order.

$$V = I_{2n}$$
while count $< n(2n - 1)$
  for $i = 1 : n$
    for $j = i + 1 : 2n$
      if $j \leq n$
        \begin{align*}
        &Q = \text{One-sided Complex Double}(A, i, j, \text{sort}, \text{tol}) \\
        &A = AQ; V = VQ
        \end{align*}
      elseif $j = n + i$
        \begin{align*}
        &Q = \text{One-sided Complex Rotation}(A, i, \text{sort}, \text{tol}) \\
        &A = AQ; V = VQ
        \end{align*}
      else
        \begin{align*}
        &Q = \text{One-sided Complex Concentric}(A, i, j, \text{sort}, \text{tol}) \\
        &A = AQ; V = VQ
        \end{align*}
    endif
  endfor
endfor
for $i = n + 1 : 2n - 1$
  for $j = i + 1 : 2n$
    $Q = \text{One-sided Complex Double}(A, i, j, 0, \text{tol})$
    $A = AQ; V = VQ$
  endfor
endfor
endwhile
for $i = 1 : 2n$
  $\sigma_i = \|A(:,i)\|; \quad U(:,i) = \frac{A(:,i)}{\sigma_i}$
endfor

Algorithm 18 ($O_{c6}, O_{c5}$). Given a complex symplectic matrix $A \in \mathbb{C}^{2n \times 2n}$, a Boolean parameter "sort", and tol $> 0$, this algorithm computes the singular values $\sigma_i$, and complex symplectic unitary matrices $U$ and $V$ so that $A = U\Sigma V^*$ where $\Sigma = \text{diag}(\sigma_i)$ is real symplectic diagonal. If "sort" is zero, then the first $n$ diagonal entries of $\Sigma$ do not appear in any particular order on the diagonal. On the other hand, if "sort" is nonzero, then the first $n$ diagonal entries of $\Sigma$ appearing in decreasing order.

$$V = I_{2n}$$
while count $< n(2n - 1)$
  for $i = 1 : n - 1$
endfor
for \( j = i + 1 : n \)

\[ Q = \text{One-sided Complex Double}(A, i, j, \text{sort, tol}) \]

\[ A = AQ; \ V = VQ \]

endfor

for \( i = 1 : n \)

\[ Q = \text{One-sided Complex Rotation}(A, i, \text{sort, tol}) \]

\[ A = AQ; \ V = VQ \]

endfor

for \( i = 1 : n - 1 \)

for \( j = n + i + 1 : 2n \)

\[ Q = \text{One-sided Complex Concentric}(A, i, j, \text{sort, tol}) \]

\[ A = AQ; \ V = VQ \]

endfor

endfor

for \( j = n + 1 : 2n - 1 \)

for \( i = j - n + 1 : n \)

\[ Q = \text{One-sided Complex Concentric}(A, i, j, \text{sort, tol}) \]

\[ A = AQ; \ V = VQ \]

endfor

endfor

for \( i = n + 1 : 2n - 1 \)

for \( j = i + 1 : 2n \)

\[ Q = \text{One-sided Complex Double}(A, i, j, 0, \text{tol}) \]

\[ A = AQ; \ V = VQ \]

endfor

endwhile

for \( i = 1 : 2n \)

\[ \sigma_i = \|A(:, i)\|; \ U(:, i) = \frac{A(:, i)}{\sigma_i} \]

endfor

We finish this section by giving pseudo-code for the functions called by Algorithms 16 and 18. These functions differ from those described in Section 6.1 only in that they first need to compute certain elements of \( A^*A \). For example, function Complex Rotation in Section 6.1 and function One-sided Complex Rotation given below both need the elements in positions \((p, p), (n + p, n + p)\) and \((p, n + p)\). Function Complex Rotation needs those elements of \( A \) while the function One-sided Complex Rotation needs those elements of \( A^*A \).

**function:** \( Q = \text{One-sided Complex Rotation}(A, p, \text{sort, tol}) \)
Given a Hermitian matrix $A \in \mathbb{C}^{2n \times 2n}$, an integer $p$ that satisfies $1 \leq p \leq n$, a Boolean parameter "sort", and $\text{tol} > 0$, this algorithm computes a complex symplectic plane rotation $Q \in \mathbb{C}^{2n \times 2n}$ as in (6.13) such that the columns $p$ and $n+p$ of $AQ$ are orthogonal to each other. That is, if $X = A^*A$, then the two off-diagonal entries in positions $(p, n+p)$ and $(n+p, p)$ of $\hat{X} = Q^*A^*AQ$ are now zero. If sort $= 0$, then the order of the entries in positions $(p, p)$ and $(n+p, n+p)$ of $X$ and $\hat{X}$ is the same. If sort is nonzero, then the sorting angle is used for ordering the entries in positions $(p, p)$ and $(n+p, n+p)$ of $\hat{X}$ in decreasing order.

\[
\begin{align*}
% \text{Compute the entries (p, p), (p, n+p), and (n+p, n+p) of } A^*A \\
x_{pp} &= \|A(:, p)\|^2; \quad x_{n+p, n+p} = \|A(:, n+p)\|^2; \quad x_{p, n+p} = (A(:, p))^*(A(:, n+p)) \\
\text{if } |x_{p, n+p}| > \text{tol} \text{ then } & \quad \% \text{compute small angle} \\
\text{if } x_{pp} \neq x_{n+p, n+p} \text{ then} \\
\tau &= \frac{x_{n+p, n+p} - x_{pp}}{2|x_{p, n+p}|} \\
t &= \frac{\text{sign}(\tau)}{|\tau| + \sqrt{\tau^2 + 1}} \\
\text{if (sort $\neq 0$) and } (x_{pp} < x_{n+p, n+p}) \text{ then} \\
t &= -\frac{1}{t} \quad \% \text{use large angle} \\
\text{endif} \\
c &= \frac{1}{\sqrt{t^2 + 1}} \quad ; \quad s = tc \\
\text{else} \\
c &= \frac{1}{\sqrt{2}} \quad ; \quad s = \frac{1}{\sqrt{2}} \\
\text{endif} \\
\alpha &= \frac{\phi}{2} \quad \% \phi \text{ denotes the angle of } x_{p, n+p} \\
Q &= I_{2n}; \quad Q(p, p) = e^{i\alpha}c; \quad Q(p, n+p) = e^{i\alpha}s \\
Q(n+p, p) &= -e^{-i\alpha}s; \quad Q(n+p, n+p) = e^{-i\alpha}c \\
\text{else} \\
Q &= I_{2n} \\
\text{endif}
\end{align*}
\]

function: $Q = \text{One-sided Complex Double}(A, p, q, \text{sort}, \text{tol})$

Given a Hermitian matrix $A \in \mathbb{C}^{2n \times 2n}$, integers $p$ and $q$ that satisfy $1 \leq p < q \leq n$, a Boolean parameter "sort", and $\text{tol} > 0$, this algorithm computes a complex symplectic double rotation $Q \in \mathbb{C}^{2n \times 2n}$ as in (6.14) such that the columns $p$ and $q$ of $AQ$ are orthogonal to each other. That is, if $X = A^*A$, then the two off-diagonal entries in positions $(p, q)$ and $(q, p)$ of $\hat{X} = Q^*A^*AQ$ are now zero. If sort =
0, then the order of the entries in positions \((p,p)\) and \((q,q)\) of \(X\) and \(\tilde{X}\) is the same. If \(\text{sort}\) is nonzero, then the sorting angle is used for ordering the entries in positions \((p,p)\) and \((q,q)\) of \(\tilde{X}\) in decreasing order.

\[
x_{pp} = \|A(:,p)\|^2; \quad x_{qq} = \|A(:,q)\|^2; \quad x_{pq} = (A(:,p))^{*}(A(:,q))
\]

if \(|x_{pq}| > \text{tol}\) then % compute small angle
    if \(x_{pp} \neq x_{qq}\) then
        \(\tau = \frac{x_{qq} - x_{pp}}{2|x_{pq}|}\)
        \(t = \frac{\text{sign} \tau}{|\tau| + \sqrt{\tau^2 + 1}}\)
    endif
    if (\(\text{sort} \neq 0\)) and \((x_{pp} < x_{qq})\) then
        \(t = -\frac{1}{t}\) % use large angle
    endif
    \(c = \frac{1}{\sqrt{t^2 + 1}}\); \(s = tc\)
else
    \(c = \frac{1}{\sqrt{2}}\); \(s = \frac{1}{\sqrt{2}}\)
endif
\(\alpha = \frac{\phi}{2}\) % \(\phi\) denotes the angle of \(x_{pq}\)
\(Q = I_{2n}; \quad Q(p,p) = e^{i\alpha}c; \quad Q(p,q) = e^{i\alpha}s\)
\(Q(q,p) = -e^{-i\alpha}s; \quad Q(q,q) = e^{-i\alpha}c\)
\(Q(n+p,n+p) = e^{i\alpha}c; \quad Q(n+p,n+q) = e^{i\alpha}s\)
\(Q(n+q,n+p) = -e^{-i\alpha}s; \quad Q(n+q,n+q) = e^{-i\alpha}c\)
else
    \(Q = I_{2n}\)
endif

Note that an algorithm for orthogonalizing columns \(n+p\) and \(n+q\) can be obtained by replacing \(p\) and \(q\) with \(n+p\) and \(n+q\), respectively in function One-sided Complex Double.

function: \(Q = \text{One-sided Complex Concentric}(A, p, n + q, \text{sort, tol})\)
Given a Hermitian matrix \(A \in \mathbb{C}^{2n \times 2n}\), and integers \(p\) and \(n + q\) that satisfy \(1 \leq p < q \leq n\), a Boolean parameter "sort”, and \(\text{tol} > 0\), this algorithm computes a complex symplectic concentric rotation \(Q \in \mathbb{C}^{2n \times 2n}\) as in (6.15) such that the
columns \( p \) and \( n+q \) of \( AQ \) are orthogonal to each other. That is, if \( X = A^*A \), then the two off-diagonal entries in position \((p, n+q)\) and \((n+q, p)\) of \( \hat{X} = Q^*A^*AQ \).

If \( \text{sort} = 0 \), then the order of the entries in positions \((p, p)\) and \((n+q, n+q)\) of \( X \) and \( \hat{X} \) is the same. If \( \text{sort} \) is nonzero, then the sorting angle is used for ordering the entries in positions \((p, p)\) and \((n+q, n+q)\) of \( X \) in decreasing order.

\[
\begin{align*}
% \text{Compute the entries} \,(p,p),\,(p,n+q),\text{and}(n+q,n+q)\text{of} \,A^*A \\
x_{pp} &= \|A(:,p)\|^2; \quad x_{n+q,n+q} = \|A(:,n+q)\|^2; \quad x_{p,n+q} = (A(:,p))^*(A(:,n+q)) \\
\text{if} \,|x_{p,n+q}| > \text{tol} \text{ then} & \% \text{compute small angle} \\
\text{if} \,x_{pp} \neq x_{n+q,n+q} \text{ then} \\
\tau &= \frac{x_{n+q,n+q} - x_{pp}}{2|x_{p,n+q}|} \\
t &= \frac{\text{sign} \,\tau}{|\tau| + \sqrt{\tau^2 + 1}} \\
\text{if} \,(\text{sort} \neq 0) \text{ and} \,(x_{pp} < x_{n+q,n+q}) \text{ then} \\
t &= -\frac{1}{t} \% \text{use large angle} \\
\text{endif} \\
c &= \frac{1}{\sqrt{t^2 + 1}}; \quad s = tc \\
\text{else} \\
c &= \frac{1}{\sqrt{2}}; \quad s = \frac{1}{\sqrt{2}} \\
\text{endif} \\
\alpha = \frac{\phi}{2} \% \phi \text{denotes the angle of} \,x_{p,n+q} \\
Q &= I_{2n}; \quad Q(p,p) = e^{i\alpha}c; \quad Q(p,n+q) = e^{i\alpha}s \\
Q(n+q,p) = -e^{-i\alpha}s; \quad Q(n+q,n+q) = e^{-i\alpha}c \\
Q(q,q) = e^{i\alpha}c; \quad Q(q,n+p) = e^{i\alpha}s \\
Q(n+p,q) = -e^{-i\alpha}s; \quad Q(n+p,n+p) = e^{-i\alpha}c \\
\text{else} \\
Q &= I_{2n} \\
\text{endif} \\
\]

Note that an algorithm for orthogonalizing columns \( q \) and \( n+p \) can be obtained by replacing \( p \) and \( n+q \) with \( q \) and \( n+p \), respectively in function One-sided Complex Concentric.
6.2.1 Numerical Experiments

In this section we present the results of numerical experiments for the one-sided Jacobi algorithms described in Section 6.2. Here

\[ A \approx U \Sigma V^* \]  

(6.20)
is a complex symplectic SVD of \( A \) computed by our algorithms, where \( U \) and \( V \) are the computed complex symplectic unitary matrices and \( \Sigma \) is the computed real symplectic diagonal matrix.

In the first set of experiments, 50 random complex symplectic matrices of dimension \( 2n \times 2n \) for \( n = 30, 40, 50, 60, 70 \), with condition number 10 were generated using MATLAB program written by N.Mackey given in [24]. All six algorithms: \( OCS_4 \), \( OCS_{4s} \), \( OCS_5 \), \( OCS_{5s} \), \( OCS_6 \), \( OCS_{6s} \), were tested on each of these matrices and the results were compared. Only the average results are reported in Tables 6.12 - 6.17 and discussed below.

- **Sweeps** denotes the average number of sweeps needed by each strategy before the algorithm is terminated. The results are similar to the real case. First, as expected, the number of sweeps increases with increasing matrix sizes for all algorithms. Second, the number of sweeps for algorithms with the sorting angle is always less than that for corresponding algorithms with the small angle. As we see from Tables 6.12 - 6.17, the average number of sweeps needed by algorithms \( OCS_4 \) and \( OCS_{4s} \) is higher than that of the other four algorithms. However, we remind the reader that the cost per full sweep of \( OCS_4 \) and \( OCS_{4s} \) is approximately half of that of \( OS_5 \), \( OS_{5s} \), \( OS_6 \), \( OS_{6s} \) as discussed in the real case.

- The symplecticity of the test matrix \( A \) was tested via \( \| AA^* - I \|_F \). It can be seen from Tables 6.12 - 6.17 that this quantity was on average, of the order \( 10^{-14} \), so our algorithms were tested on well structured matrices for all dimensions under consideration.

- \( \| UU^* - I \|_F \) and \( \| UU^* - I \|_F \) measure the average deviation of the computed left singular vectors of test matrix from symplecticity and unitarity, respectively. As we see, for all strategies and all dimensions, these quantities were, on average, of the order \( 10^{-13} \).

- \( \| VV^* - I \|_F \) and \( \| VV^* - I \|_F \) measure the average deviation of the computed right singular vectors of test matrix from symplecticity and unitarity, respectively. The computed transformations \( V \) were both complex symplectic as well as unitary to within order \( 10^{-13} \).

- The row labeled diag.alg reports the average deviation of the computed matrix \( \Sigma \) from being real symplectic diagonal, that is, the deviation from
<table>
<thead>
<tr>
<th>2n</th>
<th>60</th>
<th>80</th>
<th>100</th>
<th>120</th>
<th>140</th>
</tr>
</thead>
<tbody>
<tr>
<td>sweeps</td>
<td>9.33</td>
<td>9.78</td>
<td>10.84</td>
<td>11.40</td>
<td>11.70</td>
</tr>
<tr>
<td>$| AA^* - I |_F$</td>
<td>$1.86 \times 10^{-14}$</td>
<td>$2.71 \times 10^{-14}$</td>
<td>$3.09 \times 10^{-14}$</td>
<td>$3.73 \times 10^{-14}$</td>
<td>$4.22 \times 10^{-14}$</td>
</tr>
<tr>
<td>$| UU^* - I |_F$</td>
<td>$4.46 \times 10^{-14}$</td>
<td>$6.14 \times 10^{-14}$</td>
<td>$7.44 \times 10^{-14}$</td>
<td>$9.21 \times 10^{-14}$</td>
<td>$1.09 \times 10^{-13}$</td>
</tr>
<tr>
<td>$| UU^* - I |_F$</td>
<td>$4.97 \times 10^{-14}$</td>
<td>$2.44 \times 10^{-14}$</td>
<td>$3.74 \times 10^{-13}$</td>
<td>$5.40 \times 10^{-13}$</td>
<td>$7.11 \times 10^{-13}$</td>
</tr>
<tr>
<td>$| VV^* - I |_F$</td>
<td>$7.76 \times 10^{-14}$</td>
<td>$1.19 \times 10^{-13}$</td>
<td>$1.78 \times 10^{-13}$</td>
<td>$2.32 \times 10^{-13}$</td>
<td>$2.99 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$1.48 \times 10^{-14}$</td>
<td>$1.93 \times 10^{-14}$</td>
<td>$2.68 \times 10^{-14}$</td>
<td>$3.04 \times 10^{-14}$</td>
<td>$3.38 \times 10^{-14}$</td>
</tr>
<tr>
<td>diag.Matlab</td>
<td>$3.39 \times 10^{-15}$</td>
<td>$4.31 \times 10^{-15}$</td>
<td>$4.96 \times 10^{-15}$</td>
<td>$4.67 \times 10^{-15}$</td>
<td>$5.17 \times 10^{-15}$</td>
</tr>
<tr>
<td>rel.singular</td>
<td>$8.27 \times 10^{-15}$</td>
<td>$1.06 \times 10^{-14}$</td>
<td>$1.46 \times 10^{-14}$</td>
<td>$1.68 \times 10^{-14}$</td>
<td>$1.83 \times 10^{-14}$</td>
</tr>
</tbody>
</table>

Table 6.12: Complex One-sided Jacobi algorithm: $O_{Cs}$

<table>
<thead>
<tr>
<th>2n</th>
<th>60</th>
<th>80</th>
<th>100</th>
<th>120</th>
<th>140</th>
</tr>
</thead>
<tbody>
<tr>
<td>sweeps</td>
<td>8.11</td>
<td>8.60</td>
<td>9.08</td>
<td>9.20</td>
<td>9.35</td>
</tr>
<tr>
<td>$| AA^* - I |_F$</td>
<td>$1.86 \times 10^{-14}$</td>
<td>$2.71 \times 10^{-14}$</td>
<td>$3.09 \times 10^{-14}$</td>
<td>$3.73 \times 10^{-14}$</td>
<td>$4.22 \times 10^{-14}$</td>
</tr>
<tr>
<td>$| UU^* - I |_F$</td>
<td>$3.81 \times 10^{-14}$</td>
<td>$5.23 \times 10^{-14}$</td>
<td>$6.30 \times 10^{-14}$</td>
<td>$7.81 \times 10^{-14}$</td>
<td>$9.11 \times 10^{-13}$</td>
</tr>
<tr>
<td>$| UU^* - I |_F$</td>
<td>$4.18 \times 10^{-14}$</td>
<td>$2.17 \times 10^{-13}$</td>
<td>$3.11 \times 10^{-13}$</td>
<td>$4.48 \times 10^{-13}$</td>
<td>$6.73 \times 10^{-13}$</td>
</tr>
<tr>
<td>$| VV^* - I |_F$</td>
<td>$7.77 \times 10^{-14}$</td>
<td>$1.21 \times 10^{-13}$</td>
<td>$1.71 \times 10^{-13}$</td>
<td>$2.32 \times 10^{-13}$</td>
<td>$2.87 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$1.43 \times 10^{-14}$</td>
<td>$1.93 \times 10^{-14}$</td>
<td>$2.38 \times 10^{-14}$</td>
<td>$2.86 \times 10^{-14}$</td>
<td>$3.31 \times 10^{-14}$</td>
</tr>
<tr>
<td>diag.Matlab</td>
<td>$3.39 \times 10^{-15}$</td>
<td>$4.31 \times 10^{-15}$</td>
<td>$4.96 \times 10^{-15}$</td>
<td>$4.67 \times 10^{-15}$</td>
<td>$5.17 \times 10^{-15}$</td>
</tr>
<tr>
<td>rel.singular</td>
<td>$8.20 \times 10^{-15}$</td>
<td>$1.08 \times 10^{-14}$</td>
<td>$1.34 \times 10^{-14}$</td>
<td>$1.59 \times 10^{-14}$</td>
<td>$1.77 \times 10^{-14}$</td>
</tr>
</tbody>
</table>

Table 6.13: Complex One-sided Jacobi algorithm: $O_{Cs_{4a}}$

<table>
<thead>
<tr>
<th>2n</th>
<th>60</th>
<th>80</th>
<th>100</th>
<th>120</th>
<th>140</th>
</tr>
</thead>
<tbody>
<tr>
<td>sweeps</td>
<td>8.20</td>
<td>8.24</td>
<td>8.28</td>
<td>8.40</td>
<td>8.44</td>
</tr>
<tr>
<td>$| AA^* - I |_F$</td>
<td>$1.86 \times 10^{-14}$</td>
<td>$2.71 \times 10^{-14}$</td>
<td>$3.09 \times 10^{-14}$</td>
<td>$3.73 \times 10^{-14}$</td>
<td>$4.22 \times 10^{-14}$</td>
</tr>
<tr>
<td>$| UU^* - I |_F$</td>
<td>$4.74 \times 10^{-14}$</td>
<td>$6.42 \times 10^{-14}$</td>
<td>$8.24 \times 10^{-14}$</td>
<td>$1.02 \times 10^{-14}$</td>
<td>$1.17 \times 10^{-13}$</td>
</tr>
<tr>
<td>$| UU^* - I |_F$</td>
<td>$1.49 \times 10^{-13}$</td>
<td>$2.63 \times 10^{-13}$</td>
<td>$4.26 \times 10^{-13}$</td>
<td>$6.09 \times 10^{-13}$</td>
<td>$7.99 \times 10^{-13}$</td>
</tr>
<tr>
<td>$| VV^* - I |_F$</td>
<td>$1.10 \times 10^{-13}$</td>
<td>$1.70 \times 10^{-13}$</td>
<td>$2.54 \times 10^{-13}$</td>
<td>$3.30 \times 10^{-13}$</td>
<td>$4.30 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$2.05 \times 10^{-14}$</td>
<td>$2.16 \times 10^{-14}$</td>
<td>$3.67 \times 10^{-14}$</td>
<td>$4.18 \times 10^{-14}$</td>
<td>$5.17 \times 10^{-14}$</td>
</tr>
<tr>
<td>diag.Matlab</td>
<td>$3.39 \times 10^{-15}$</td>
<td>$4.31 \times 10^{-15}$</td>
<td>$4.96 \times 10^{-15}$</td>
<td>$4.67 \times 10^{-15}$</td>
<td>$5.17 \times 10^{-15}$</td>
</tr>
<tr>
<td>rel.singular</td>
<td>$1.11 \times 10^{-14}$</td>
<td>$1.16 \times 10^{-14}$</td>
<td>$1.97 \times 10^{-14}$</td>
<td>$2.18 \times 10^{-14}$</td>
<td>$2.68 \times 10^{-14}$</td>
</tr>
</tbody>
</table>

Table 6.14: Complex One-sided Jacobi algorithm: $O_{Cs_{5}}$
Table 6.15: Complex One-sided Jacobi algorithm: $O_{C S_{55}}$

<table>
<thead>
<tr>
<th>2n sweeps</th>
<th>60</th>
<th>80</th>
<th>100</th>
<th>120</th>
<th>140</th>
</tr>
</thead>
<tbody>
<tr>
<td>$|AA^* - I|_F$</td>
<td>$1.86 \times 10^{-14}$</td>
<td>$2.71 \times 10^{-14}$</td>
<td>$3.09 \times 10^{-14}$</td>
<td>$3.73 \times 10^{-14}$</td>
<td>$4.22 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|UU^* - I|_F$</td>
<td>$4.03 \times 10^{-14}$</td>
<td>$5.35 \times 10^{-14}$</td>
<td>$7.05 \times 10^{-14}$</td>
<td>$8.52 \times 10^{-14}$</td>
<td>$9.91 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|UU^* - I|_F$</td>
<td>$1.65 \times 10^{-13}$</td>
<td>$3.07 \times 10^{-13}$</td>
<td>$4.48 \times 10^{-13}$</td>
<td>$6.66 \times 10^{-13}$</td>
<td>$8.80 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|VV^* - I|_F$</td>
<td>$9.10 \times 10^{-14}$</td>
<td>$1.39 \times 10^{-13}$</td>
<td>$2.07 \times 10^{-13}$</td>
<td>$2.72 \times 10^{-13}$</td>
<td>$3.56 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|VV^* - I|_F$</td>
<td>$9.12 \times 10^{-14}$</td>
<td>$1.39 \times 10^{-13}$</td>
<td>$2.07 \times 10^{-13}$</td>
<td>$2.73 \times 10^{-13}$</td>
<td>$3.56 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$1.66 \times 10^{-14}$</td>
<td>$2.12 \times 10^{-14}$</td>
<td>$2.80 \times 10^{-14}$</td>
<td>$3.22 \times 10^{-14}$</td>
<td>$4.02 \times 10^{-14}$</td>
</tr>
<tr>
<td>diag.Matlab</td>
<td>$3.39 \times 10^{-15}$</td>
<td>$4.31 \times 10^{-15}$</td>
<td>$4.96 \times 10^{-15}$</td>
<td>$4.67 \times 10^{-15}$</td>
<td>$5.17 \times 10^{-15}$</td>
</tr>
<tr>
<td>rel.singular</td>
<td>$9.16 \times 10^{-14}$</td>
<td>$1.21 \times 10^{-14}$</td>
<td>$1.52 \times 10^{-14}$</td>
<td>$1.77 \times 10^{-14}$</td>
<td>$2.13 \times 10^{-14}$</td>
</tr>
</tbody>
</table>

Table 6.16: Complex One-sided Jacobi algorithm: $O_{C S_{66}}$

<table>
<thead>
<tr>
<th>2n sweeps</th>
<th>60</th>
<th>80</th>
<th>100</th>
<th>120</th>
<th>140</th>
</tr>
</thead>
<tbody>
<tr>
<td>$|AA^* - I|_F$</td>
<td>$1.86 \times 10^{-14}$</td>
<td>$2.71 \times 10^{-14}$</td>
<td>$3.09 \times 10^{-14}$</td>
<td>$3.73 \times 10^{-14}$</td>
<td>$4.22 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|UU^* - I|_F$</td>
<td>$4.54 \times 10^{-14}$</td>
<td>$5.89 \times 10^{-14}$</td>
<td>$7.79 \times 10^{-14}$</td>
<td>$9.44 \times 10^{-14}$</td>
<td>$1.10 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|UU^* - I|_F$</td>
<td>$1.22 \times 10^{-13}$</td>
<td>$2.26 \times 10^{-13}$</td>
<td>$4.12 \times 10^{-13}$</td>
<td>$5.83 \times 10^{-13}$</td>
<td>$7.78 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|VV^* - I|_F$</td>
<td>$9.72 \times 10^{-14}$</td>
<td>$1.50 \times 10^{-13}$</td>
<td>$2.17 \times 10^{-13}$</td>
<td>$2.89 \times 10^{-13}$</td>
<td>$3.73 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|VV^* - I|_F$</td>
<td>$9.71 \times 10^{-14}$</td>
<td>$1.50 \times 10^{-13}$</td>
<td>$2.17 \times 10^{-13}$</td>
<td>$2.89 \times 10^{-13}$</td>
<td>$3.73 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$1.78 \times 10^{-14}$</td>
<td>$2.42 \times 10^{-14}$</td>
<td>$3.00 \times 10^{-14}$</td>
<td>$3.59 \times 10^{-14}$</td>
<td>$4.18 \times 10^{-14}$</td>
</tr>
<tr>
<td>diag.Matlab</td>
<td>$3.39 \times 10^{-15}$</td>
<td>$4.31 \times 10^{-15}$</td>
<td>$4.96 \times 10^{-15}$</td>
<td>$4.67 \times 10^{-15}$</td>
<td>$5.17 \times 10^{-15}$</td>
</tr>
<tr>
<td>rel.singular</td>
<td>$1.04 \times 10^{-14}$</td>
<td>$1.29 \times 10^{-14}$</td>
<td>$1.60 \times 10^{-14}$</td>
<td>$1.92 \times 10^{-14}$</td>
<td>$2.20 \times 10^{-14}$</td>
</tr>
</tbody>
</table>

Table 6.17: Complex One-sided Jacobi algorithm: $O_{C S_{66}}$

<table>
<thead>
<tr>
<th>2n sweeps</th>
<th>60</th>
<th>80</th>
<th>100</th>
<th>120</th>
<th>140</th>
</tr>
</thead>
<tbody>
<tr>
<td>$|AA^* - I|_F$</td>
<td>$1.86 \times 10^{-14}$</td>
<td>$2.71 \times 10^{-14}$</td>
<td>$3.09 \times 10^{-14}$</td>
<td>$3.73 \times 10^{-14}$</td>
<td>$4.22 \times 10^{-14}$</td>
</tr>
<tr>
<td>$|UU^* - I|_F$</td>
<td>$4.23 \times 10^{-14}$</td>
<td>$5.67 \times 10^{-14}$</td>
<td>$7.12 \times 10^{-14}$</td>
<td>$8.95 \times 10^{-14}$</td>
<td>$1.02 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|UU^* - I|_F$</td>
<td>$1.40 \times 10^{-13}$</td>
<td>$2.53 \times 10^{-13}$</td>
<td>$4.12 \times 10^{-13}$</td>
<td>$4.26 \times 10^{-13}$</td>
<td>$7.06 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|VV^* - I|_F$</td>
<td>$1.01 \times 10^{-13}$</td>
<td>$1.52 \times 10^{-13}$</td>
<td>$2.26 \times 10^{-13}$</td>
<td>$2.83 \times 10^{-13}$</td>
<td>$3.99 \times 10^{-13}$</td>
</tr>
<tr>
<td>$|VV^* - I|_F$</td>
<td>$1.01 \times 10^{-13}$</td>
<td>$1.52 \times 10^{-13}$</td>
<td>$2.26 \times 10^{-13}$</td>
<td>$2.83 \times 10^{-13}$</td>
<td>$3.99 \times 10^{-13}$</td>
</tr>
<tr>
<td>diag.alg</td>
<td>$1.81 \times 10^{-14}$</td>
<td>$2.23 \times 10^{-14}$</td>
<td>$2.90 \times 10^{-14}$</td>
<td>$3.28 \times 10^{-14}$</td>
<td>$4.37 \times 10^{-14}$</td>
</tr>
<tr>
<td>diag.Matlab</td>
<td>$3.39 \times 10^{-15}$</td>
<td>$4.31 \times 10^{-15}$</td>
<td>$4.96 \times 10^{-15}$</td>
<td>$4.67 \times 10^{-15}$</td>
<td>$5.17 \times 10^{-15}$</td>
</tr>
<tr>
<td>rel.singular</td>
<td>$1.04 \times 10^{-14}$</td>
<td>$1.22 \times 10^{-14}$</td>
<td>$1.58 \times 10^{-14}$</td>
<td>$1.77 \times 10^{-14}$</td>
<td>$2.31 \times 10^{-14}$</td>
</tr>
</tbody>
</table>
pairing structure \((\sigma, 1/\sigma)\) between the singular values computed by our algorithms. Here \(\text{diag.alg} = \max\{x \ast y - 1\}\) where \(x = [\sigma_1, \sigma_2, \ldots, \sigma_n]\), \(y = [\sigma_{n+1}, \sigma_{n+2}, \ldots, \sigma_{2n}]\), and \(\ast\) denotes array multiplication (as in MATLAB). Similarly, the row labeled \(\text{diag.Matlab}\) reports the deviation from pairing structure of the singular values computed by MATLAB’s \text{svd} function. If \(v\) denotes the vector of sorted singular values computed by \text{svd}(A), and \(x := v(1 : n), y := v(2n : -1 : n + 1)\), then \(\text{diag.Matlab} = \max\{x \ast y - 1\}\). As we see from Tables 6.12 - 6.17, the loss of the symplectic structure of \(\Sigma\) was, on average, of the order \(10^{-14}\) while the loss of pairing structure of singular values computed by MATLAB was of the order \(10^{-15}\) for all algorithms and all dimensions under consideration.

- The row labeled \(\text{rel.singular}\) records the maximum relative error as measured by

\[
\text{rel.singular} = \max_j \left| \frac{\sigma_{j}^{\text{mat}} - \sigma_{j}^{\text{alg}}}{\sigma_{j}^{\text{mat}}} \right|
\]

where \(\{\sigma_{j}^{\text{mat}}\}\) and \(\{\sigma_{j}^{\text{alg}}\}\) denote the singular values computed by MATLAB’s \text{svd} function and by our algorithms, respectively. As can be seen, the singular values computed by our strategies when compared to MATLAB had relative error around \(10^{-14}\), for all dimensions under consideration.

A second set of experiments was designed to see how accurately our algorithms computed the singular values of a complex symplectic test matrix with pre-chosen singular values. Then we compared the known singular values to singular values computed by our strategies and to those computed by MATLAB’s \text{svd} function. Some of the experiments are shown below.

**Example 18.** A complex symplectic matrix \(A\) was generated with singular values 5, 5, 3, \(\frac{1}{3}, \frac{1}{3}\). In this case \(A\) is well-conditioned with condition number 25, and has repeated singular values of multiplicity 2. Table 6.18 shows the exact singular values and the relative errors in the singular value approximations computed by our algorithms and by MATLAB’s \text{svd} function.

**Example 19.** A complex symplectic matrix \(A\) was generated with singular values \(a, b, c, \frac{1}{3}, \frac{1}{3}\) where \([a, b, c] = [3, 3, 3] + [2.7, 3.1, 6.7] \times 10^{-15}\). Here \(A\) is well-conditioned and has two clusters of singular values of “multiplicity” 3 — singular values near 3 and near \(\frac{1}{3}\). The exact singular values and the relative errors are given in Table 6.19.

The results from Examples 18 and 19 as shown in Tables 6.18 and 6.19 show that our algorithms yield the singular values with maximum relative error of order \(\text{eps}\) which is the same as MATLAB’s \text{svd} function when the generated matrix \(A\) is well-conditioned.

143
<table>
<thead>
<tr>
<th>$\sigma_j$</th>
<th>$O_C S_4$</th>
<th>$O_C S_{4a}$</th>
<th>$O_C S_5$</th>
<th>$O_C S_{5a}$</th>
<th>$O_C S_6$</th>
<th>$O_C S_{6a}$</th>
<th>Matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/5</td>
<td>0</td>
<td>2.8e-16</td>
<td>0</td>
<td>4.1e-16</td>
<td>0</td>
<td>9.7e-16</td>
<td>2.7e-16</td>
</tr>
<tr>
<td>1/5</td>
<td>4.1e-16</td>
<td>6.9e-16</td>
<td>1.1e-15</td>
<td>4.1e-16</td>
<td>1.1e-15</td>
<td>1.2e-15</td>
<td>2.7e-16</td>
</tr>
<tr>
<td>1/3</td>
<td>9.9e-16</td>
<td>5.0e-16</td>
<td>3.3e-16</td>
<td>1.6e-15</td>
<td>0</td>
<td>9.9e-16</td>
<td>8.3e-16</td>
</tr>
<tr>
<td>3</td>
<td>3.0e-16</td>
<td>0</td>
<td>7.4e-16</td>
<td>1.4e-15</td>
<td>1.4e-16</td>
<td>2.9e-16</td>
<td>2.9e-16</td>
</tr>
<tr>
<td>5</td>
<td>1.8e-16</td>
<td>0</td>
<td>1.7e-16</td>
<td>7.1e-16</td>
<td>1.7e-16</td>
<td>3.5e-16</td>
<td>5.3e-16</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1.8e-16</td>
<td>1.7e-16</td>
<td>0</td>
<td>1.7e-16</td>
<td>1.7e-16</td>
<td>3.5e-16</td>
</tr>
</tbody>
</table>

Table 6.18: Accuracy of the singular values $5, 5, 3, \frac{1}{5}, \frac{1}{5}, \frac{1}{3}$

<table>
<thead>
<tr>
<th>$\sigma_j$</th>
<th>$O_C S_4$</th>
<th>$O_C S_{4a}$</th>
<th>$O_C S_5$</th>
<th>$O_C S_{5a}$</th>
<th>$O_C S_6$</th>
<th>$O_C S_{6a}$</th>
<th>Matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/c</td>
<td>0</td>
<td>1.7e-16</td>
<td>9.9e-16</td>
<td>5.0e-16</td>
<td>0</td>
<td>3.3e-16</td>
<td>6.6e-16</td>
</tr>
<tr>
<td>1/b</td>
<td>0</td>
<td>6.7e-16</td>
<td>0</td>
<td>6.6e-16</td>
<td>1.6e-16</td>
<td>1.6e-16</td>
<td>1.6e-16</td>
</tr>
<tr>
<td>1/a</td>
<td>8.3e-16</td>
<td>8.3e-16</td>
<td>3.3e-16</td>
<td>8.3e-16</td>
<td>5.0e-16</td>
<td>5.0e-16</td>
<td>0</td>
</tr>
<tr>
<td>a</td>
<td>0</td>
<td>0</td>
<td>4.4e-16</td>
<td>5.9e-16</td>
<td>4.4e-16</td>
<td>1.4e-16</td>
<td>5.9e-16</td>
</tr>
<tr>
<td>b</td>
<td>3.0e-16</td>
<td>3.0e-16</td>
<td>5.9e-16</td>
<td>5.9e-16</td>
<td>2.9e-16</td>
<td>2.9e-16</td>
<td>8.8e-16</td>
</tr>
<tr>
<td>c</td>
<td>4.4e-16</td>
<td>4.4e-16</td>
<td>1.4e-16</td>
<td>2.9e-16</td>
<td>2.9e-16</td>
<td>2.9e-16</td>
<td>2.9e-16</td>
</tr>
</tbody>
</table>

Table 6.19: Accuracy of the singular values near $3$ and $\frac{1}{3}$

**Example 20.** A complex symplectic matrix $A$ was generated with singular values $100, 10, 2, \frac{1}{100}, \frac{1}{10}, \frac{1}{2}$. In this case, $A$ is an ill-conditioned matrix with condition number $10^4$. The exact singular values and the relative errors are reported in Table 6.20.

<table>
<thead>
<tr>
<th>$\sigma_j$</th>
<th>$O_C S_4$</th>
<th>$O_C S_{4a}$</th>
<th>$O_C S_5$</th>
<th>$O_C S_{5a}$</th>
<th>$O_C S_6$</th>
<th>$O_C S_{6a}$</th>
<th>Matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/100</td>
<td>3.8e-14</td>
<td>3.4e-14</td>
<td>1.3e-13</td>
<td>9.1e-15</td>
<td>7.8e-14</td>
<td>6.9e-16</td>
<td>4.0e-13</td>
</tr>
<tr>
<td>1/10</td>
<td>1.1e-14</td>
<td>1.6e-14</td>
<td>1.9e-14</td>
<td>1.2e-14</td>
<td>9.5e-15</td>
<td>2.0e-14</td>
<td>9.8e-14</td>
</tr>
<tr>
<td>1/2</td>
<td>1.1e-15</td>
<td>4.0e-15</td>
<td>8.6e-16</td>
<td>1.8e-15</td>
<td>7.5e-15</td>
<td>8.9e-15</td>
<td>1.3e-15</td>
</tr>
<tr>
<td>2</td>
<td>1.1e-15</td>
<td>1.8e-15</td>
<td>2.2e-15</td>
<td>5.5e-16</td>
<td>3.0e-15</td>
<td>2.0e-15</td>
<td>1.7e-15</td>
</tr>
<tr>
<td>10</td>
<td>3.6e-16</td>
<td>1.8e-15</td>
<td>1.7e-16</td>
<td>1.7e-16</td>
<td>1.7e-16</td>
<td>8.8e-16</td>
<td>5.3e-16</td>
</tr>
<tr>
<td>100</td>
<td>2.8e-16</td>
<td>5.7e-16</td>
<td>5.6e-16</td>
<td>5.6e-16</td>
<td>8.5e-16</td>
<td>1.4e-15</td>
<td>4.2e-16</td>
</tr>
</tbody>
</table>

Table 6.20: Accuracy of the singular values $100, 10, 2, \frac{1}{100}, \frac{1}{10}, \frac{1}{2}$

As we see from this table, the singular values computed by our strategies and by MATLAB are comparable. However, our algorithms compute the small singular values with more high relative accuracy than MATLAB's `svd` function.

**Example 21.** A $20 \times 20$ complex symplectic matrix $A$ was generated with singular values $2, \frac{1}{2}, 4, \frac{1}{4}, 6, \frac{1}{6}, 7, \frac{1}{7}, a, b, c, \frac{1}{a}, \frac{1}{b}, \frac{1}{c}, x, y, z, \frac{1}{x}, \frac{1}{y}, \frac{1}{z}$ where $[a, b, c] = [3, 3, 3] + [0.4, 2.7, 8.9] \times 10^{-15}$ and $[x, y, z] = [5, 5, 5] + [0.9, 1.8, 3.6] \times 10^{-15}$. In this case,
A is well-conditioned and has 4 clusters of singular values of "multiplicity" 3 —
singular values near 3, \( \frac{1}{3} \), 5, and \( \frac{1}{5} \). Table 6.21 reports the selected singular values
and the relative errors in the singular value computed by our algorithms and by
MATLAB’s \( \texttt{svd} \) function for each set of clusters.

<table>
<thead>
<tr>
<th>( \sigma_j )</th>
<th>( O_{CS_4} )</th>
<th>( O_{CS_{s4}} )</th>
<th>( O_{CS_5} )</th>
<th>( O_{CS_{s5}} )</th>
<th>( O_{CS_6} )</th>
<th>( O_{CS_{s6}} )</th>
<th>Matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1.2e-15</td>
<td>1.5e-15</td>
<td>4.8e-15</td>
<td>4.8e-15</td>
<td>4.8e-15</td>
<td>3.1e-15</td>
<td>2.2e-15</td>
</tr>
<tr>
<td>b</td>
<td>3.0e-16</td>
<td>8.9e-16</td>
<td>6.6e-15</td>
<td>5.3e-15</td>
<td>4.0e-15</td>
<td>2.2e-15</td>
<td>0</td>
</tr>
<tr>
<td>c</td>
<td>1.0e-15</td>
<td>1.5e-16</td>
<td>2.2e-15</td>
<td>1.7e-15</td>
<td>1.7e-15</td>
<td>8.8e-15</td>
<td>0</td>
</tr>
<tr>
<td>1/a</td>
<td>5.0e-16</td>
<td>5.0e-16</td>
<td>2.7e-16</td>
<td>7.7e-16</td>
<td>6.0e-16</td>
<td>1.6e-16</td>
<td>1.6e-16</td>
</tr>
<tr>
<td>1/b</td>
<td>5.0e-16</td>
<td>1.7e-15</td>
<td>4.4e-16</td>
<td>0</td>
<td>1.1e-16</td>
<td>5.5e-16</td>
<td>3.3e-16</td>
</tr>
<tr>
<td>1/c</td>
<td>0</td>
<td>1.7e-15</td>
<td>8.3e-16</td>
<td>6.6e-16</td>
<td>6.1e-16</td>
<td>0</td>
<td>2.2e-16</td>
</tr>
<tr>
<td>x</td>
<td>1.2e-15</td>
<td>0</td>
<td>5.3e-15</td>
<td>4.4e-15</td>
<td>8.8e-15</td>
<td>2.6e-15</td>
<td>3.5e-15</td>
</tr>
<tr>
<td>y</td>
<td>0</td>
<td>5.3e-16</td>
<td>5.3e-15</td>
<td>5.3e-15</td>
<td>8.8e-15</td>
<td>0</td>
<td>1.7e-15</td>
</tr>
<tr>
<td>z</td>
<td>5.3e-16</td>
<td>1.2e-15</td>
<td>5.3e-15</td>
<td>5.3e-15</td>
<td>1.1e-14</td>
<td>1.7e-15</td>
<td>2.6e-15</td>
</tr>
<tr>
<td>1/x</td>
<td>1.4e-16</td>
<td>5.6e-16</td>
<td>2.3e-16</td>
<td>5.2e-16</td>
<td>6.9e-16</td>
<td>5.2e-16</td>
<td>2.7e-16</td>
</tr>
<tr>
<td>1/y</td>
<td>4.2e-16</td>
<td>0</td>
<td>2.5e-16</td>
<td>3.6e-16</td>
<td>6.1e-16</td>
<td>2.7e-16</td>
<td>1.9e-16</td>
</tr>
<tr>
<td>1/z</td>
<td>3.1e-15</td>
<td>4.2e-16</td>
<td>1.6e-16</td>
<td>5.5e-17</td>
<td>2.2e-16</td>
<td>2.7e-16</td>
<td>6.6e-16</td>
</tr>
</tbody>
</table>

Table 6.21: Accuracy of the singular values near 3, \( \frac{1}{3} \), 5 and \( \frac{1}{5} \)

From Table 6.21, we see that our algorithms compute singular values with the
relative error comparable to MATLAB’s \( \texttt{svd} \) function for each cluster.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Chapter 7

Concluding Remarks

The motivation for this thesis comes from the recent result of Mackey, Mackey, and Mehrmann [25] and Xu [42] that shows every symplectic matrix has a symplectic SVD. Our goal in this thesis is to develop structure preserving algorithms that compute all three structured factors in a symplectic SVD of a symplectic matrix.

We have presented two types of structure preserving algorithms. The first algorithm uses the close relationship between the SVD and the polar decomposition. This relation reduces the problem of finding a symplectic SVD to that of calculating the structured spectral decomposition of a real symplectic symmetric or complex symplectic Hermitian matrix. A Jacobi-like method was developed to compute these doubly structured spectral decompositions.

The second algorithm was a one-sided Jacobi method that directly computes the symplectic SVD of real and complex symplectic matrices.

The structure preserving tools used by our algorithms were symplectic plane rotations and symplectic double rotations of two types — with direct sum embedding and with concentric embedding. For the real $4 \times 4$ case, we found a finite step algorithm for computing a doubly structured spectral decomposition.

The two rotations that make up a symplectic double rotation of either type can be distinguished as an active rotation and an inactive rotation. Under a similarity by a double rotation, the inactive rotation can cause some leakage of weight from the diagonal entries into the off-diagonal entries. This leads to sequence of off-diagonal norms of the matrix iterates not being a strictly decreasing sequence, and hence any proof of convergence of our algorithms cannot rely on the strategy of showing that the off-diagonal norm decreases.

For the $2n \times 2n$ case, we presented six sweep designs, called strategies $S_1, S_2, S_3, S_4, S_5, \text{ and } S_6$ for our algorithms. However, due to the large number of sweeps needed for each algorithm to converge to the desired form, $S_1, S_2, \text{ and } S_3$ were eliminated from further consideration and $S_4, S_5, \text{ and } S_6$ were chosen as the main sweep designs for all algorithms. Numerical experiments showed that...
an algorithm with the design \( S_4 \) requires more sweeps than one with the designs \( S_5 \) and \( S_6 \). However, a full sweep of design \( S_4 \) incurs approximately half the cost of a full sweep corresponding to design \( S_5 \) or \( S_6 \). Numerical experiments reported that the total number of iterations needed by algorithm with the design \( S_4 \) was lower than that with the design \( S_5 \) or \( S_6 \).

We also presented appropriate strategies for using the sorting angle for a rotation. When these strategies are used, the first \( n \) entries on the diagonal of \( \Sigma \) appear in decreasing order. Moreover, numerical experiments showed that the algorithms using the sorting angle require fewer sweeps than those using the small angle.

The same structure preserving tools and sweep designs were used to develop one-sided Jacobi algorithms that directly compute a real symplectic SVD of a real symplectic matrix.

The algorithms for the real case were extended to the complex case. Tools used by the complex algorithms were described. We first presented Jacobi-like algorithms for computing the structured spectral decomposition of a complex symplectic Hermitian matrix. A one-sided Jacobi method for directly computing the complex symplectic SVD was then presented.

Numerical experiments presented here show that our algorithms provide well structured factors of the symplectic SVD and converge quadratically.

**Future Work:** The following are interesting open problems related to and arising from this thesis.

1. As we discussed in this thesis, a proof of convergence of our algorithms cannot rely on the strategy of showing that the off-diagonal norm decreases with every iteration; proof of convergence by other means remains to be found.

2. We only presented structure preserving algorithms for computing the symplectic SVD of real symplectic matrices and complex symplectic matrices. Extending these algorithms to compute a conjugate symplectic SVD of a conjugate symplectic matrix remains to be done. Two major hurdles of an extension are a finding suitable sweep designs and appropriate tools.

3. A long standing problem is the development of a good Jacobi algorithm for computing the Hamiltonian-Schur form of a Hamiltonian matrix. Recall that complex Hamiltonian matrices form the Lie algebra associated with the bilinear form \( \langle x, y \rangle \mapsto x^T J y \). When written in matrix form, a Hamiltonian matrix \( H \in \mathbb{C}^{2n \times 2n} \) can be represented as

\[
H = \begin{bmatrix}
E & F \\
G & -E^T
\end{bmatrix},
\]

(7.1)
where $E, F, G \in \mathbb{C}^{n \times n}$, $F^T = F$, and $G^T = G$. When the form is restricted to $\mathbb{R}^{2n}$, we get the Lie algebra of real Hamiltonian matrices and the matrices in (7.1) are now all real. On the other hand, conjugate Hamiltonian matrices form the Lie algebra associated with the sesquilinear form on $\mathbb{C}^{2n}$ given by $\langle x, y \rangle \mapsto x^* J y$, and they can be represented in matrix form as

$$H = \begin{bmatrix} E & F \\ G & -E^* \end{bmatrix}, \quad (7.2)$$

where $E, F, G \in \mathbb{C}^{n \times n}$, $F^* = F$, and $G^* = G$. In [4], Byers presented a Jacobi algorithm for calculating a Hamiltonian-Schur form of a conjugate Hamiltonian matrix. Structure preserving tools in his algorithm were real symplectic plane rotations and symplectic unitary matrices built using $2 \times 2$ unitary matrices with direct sum embedding. With these tools, his algorithm reduces a conjugate Hamiltonian matrix $H$ as in (7.2) to a conjugate Hamiltonian matrix of the form

$$\begin{bmatrix} B & R \\ 0 & -B^* \end{bmatrix},$$

where $B \in \mathbb{C}^{n \times n}$ is upper triangular and $R \in \mathbb{C}^{n \times n}$ is Hermitian. However, as remarked in [4], the algorithm converges very slowly. We wish to investigate the effectiveness of using an additional tool — double rotations with concentric embedding — together with our new sweep designs in developing a viable Jacobi algorithm for computing the Hamiltonian-Schur form of Hamiltonian matrices.
Bibliography


